

Валентин Юльевич Арьков
Командная строка CMD

Учебное пособие



Валентин Арьков

**Командная строка CMD.
Учебное пособие**

«Издательские решения»

Арьков В. Ю.

Командная строка CMD. Учебное пособие / В. Ю. Арьков —
«Издательские решения»,

ISBN 978-5-00-554126-0

Изучая операционные системы, важно и понимать теорию, и уметь практически работать на компьютере. Функции любой операционной системы сводятся к понятию управления. Реализация этих функций — это выбор интерфейса пользователя: командная строка, файловый менеджер или оконная система. В курсе ОС мы рассматриваем все три вида интерфейса — практически, но с пониманием теории. Для упражнений будем использовать популярную ОС, название которой буквально переводится как «Маленькие Мягкие Окна».

ISBN 978-5-00-554126-0

© Арьков В. Ю.
© Издательские решения

Содержание

Введение	7
Задания	8
Знакомство с интерфейсом	9
Запуск командного окна через [Win + R]	10
VER – Версия ОС	12
История ввода команд	13
Полоса прокрутки	14
Запуск командного окна через поиск	15
Язык интерфейса	17
Запуск командного окна через меню	20
Настройка окна	25
TITLE – Заголовок окна	27
HELP – Получение справки	28
PROMPT – Приглашение ОС	30
START – Выполнение команды	31
Запуск CMD в выбранном каталоге	32
Другие способы запустить CMD	33
Стандартный ввод-вывод	34
TYPE – Вывод файла на экран	35
OSK – Экранная клавиатура	36
COPY CON – Работа с консолью	37
COPY NUL – «Нулевое» устройство	39
> – Перенаправление вывода	40
>> Добавление к файлу	41
SORT – сортировка	43
<- Перенаправление ввода	45
MORE – Постраничный вывод	47
– Конвейер	48
& – Последовательный запуск команд	49
stdio. h – Стандартный ввод-вывод в Си	50
Файлы и каталоги	60
DIR – Содержимое каталога	61
STDERR – Поток сообщений об ошибках	66
CD – Текущий каталог	67
TREE – Файловое дерево	70
COPY – Копирование файлов	71
MOVE – Перенести файлы	72
DEL – Удаление файла	73
RENAME – Переименование файла	74
MKDIR – Создание каталога	75
RMDIR – Удаление каталога	76
ATTRIB – Атрибуты и разрешения	77
MKLINK – Ссылка на файл	79
BAT/CMD – Пакетные файлы	81
CMD – Запуск командной строки	83
START – Запуск в новом окне	85

ECHO – Вывод сообщения	87
Очередность вызова	89
% – аргументы командной строки	90
FOR – Цикл	93
SHIFT – смещение нумерации аргументов	94
SET – Параметры окружения	95
SET – Вывод параметров	96
PATHEXT – Порядок запуска	97
PATH – Путь для поиска	98
7-ZIP – архиватор файлов	100
Загрузка программы	101
PATH – Путь к программе	103
ADD – Создание архива	104
LIST – Оглавление архива	107
TZIP – Создание ZIP-архива	108
SFX – самораспаковывающийся архив	109
Заключение	112
Литература	113

Командная строка CMD

Учебное пособие

Валентин Юльевич Арьков

© Валентин Юльевич Арьков, 2021

ISBN 978-5-0055-4126-0

Создано в интеллектуальной издательской системе Ridero

Введение

В качестве введения мы обсудим так называемый «Дисклеймер» – Disclaimer – Отказ от претензий. Это модное нынче название для предупреждения читателя о последствиях чтения книги или просмотра фильма. Иногда помогает защититься от судебных претензий, особенно если у читателя напрочь отсутствует чувство юмора.

Данное произведение предназначено исключительно для изучения операционной системы. Текст составлен на основе многолетнего опыта автора по работе в самых разных вычислительных средах, начиная с больших машин серии ЕС ЭВМ и малых вычислительных машин серии СМ, далее через мини-ЭВМ и микро-ЭВМ – вплоть до наших дней с разнообразными настольными, мобильными, персональными и супер-компьютерами.

Любые совпадения с реальными людьми, корпорациями или событиями являются случайными и непреднамеренными. Все персонажи являются вымышленными. Все факты предлагается проверить читателю и убедиться в том, что переводы иностранных слов соответствуют словарным определениям, а описанные приёмы работы применимы на практике.

Любые высказывания по поводу истории разработки программного обеспечения, качества программных продуктов или уровня перевода являются оценочными суждениями автора и не призывают к каким-либо действиям, кроме как к углублённому изучению материала и большему пониманию смысла происходящего в мире информационных технологий.

Задание. Найдите в своих «интернетах» происхождение фразы «Не стреляйте в пианиста...» Выясните, как звучит эта фраза полностью и насколько она применима к компьютерному миру.

Задания

Для освоения материала студенту предстоит изучить небольшой раздел текста и рисунки. Именно изучить, а не прочесть, и не просмотреть, и не говорить: «Это я и так знаю, а это мне точно не пригодится». Узнать что-то новое. Достичь нового понимания. Увидеть, как всё в жизни связано.

Затем нужно выполнить **задание**. Можно выполнить его несколько раз. Можно слегка изменить действия, поработать с разными настройками, «поиграться» с командами. Научиться вызывать очередную команду. Это для тех, кто хочет просто «сдать лабы».

Задание. Выясните, что такое «знания» и «умения», что у них общего и в чём разница между «знать» и «уметь».

Кроме обычных заданий, в тексте есть **творческие задания**. Это задания повышенной сложности. Это для тех, кто хочет учиться. Простые задания – это как «колбаса для населения». Просто и доступно. Сложные задания заставляют человека подумать, разобраться, поэкспериментировать. Это ещё один шаг к тому, чтобы стать специалистом своего дела.

Творческое задание. Выясните, что означает выражение «колбаса для населения».

Знакомство с интерфейсом

Самый первый вид интерфейса пользователя, с которым мы познакомимся, – это командная строка – `Command prompt`. Здесь работа идёт в текстовом режиме.

Вот некоторые другие названия для «текстового» интерфейса:

- `Command Window` – командное окно
- `Command Shell` – командная оболочка
- `Command Interpreter` —интерпретатор команд
- `Command Processor` – командный процессор или обработчик команд
- `CLI – Command Line Interface` – Интерфейс командной строки
- `Console` – Консоль

В развитие идеи командной оболочки ОС была разработана оболочка для сисадмина – **PowerShell**.

Большинство перечисленных названий говорят о том, что данный интерфейс имеет отношение к вводу КОМАНД. Здесь пользователь вводит команды с клавиатуры.

Первоначально в качестве интерфейса пользователя использовалась электрическая печатная машинка с длинным рулоном бумаги. Рулон можно прокрутить вперёд и назад, чтобы просмотреть историю работы.

Затем была реализация командной строки на экране компьютера в текстовой режиме. И это действительно была СТРОКА. Это был текст – и никакого графического интерфейса.

Современная программная реализация командной строки в виде окна (в графическом режиме) тоже имеет возможность прокрутки результатов вверх и вниз – как на рулоне бумаги.

Мы будем знакомиться с интерпретатором командной строки (командным окном) на примере популярной ОС *Microsoft Windows*, сокращённо *MS Windows*, или просто *Windows*. Здесь мы встретим самые общие принципы и подходы, которые можно найти и в любой другой операционной системе.

Название компании, а точнее, корпорации, состоит из двух частей. Слово *micro* означает «маленькая». А вот что именно у них маленькое, доверьтесь узнать читателю самостоятельно.

Задание. Просмотрите в Википедии статью **Microsoft**. Выясните, сколько человек было в этой компании первоначально и что означает название этой корпорации.

Вторая часть названия имеет отношение к программному обеспечению – *software*. Буквально слово *soft* означает «мягкий», но у него есть и другие значения.

Одно из объяснений звучит так. Первые компьютеры 1940-х годов были очень большими и содержали много тяжёлых металлических деталей. За это компьютерное оборудование прозвали «железом». По-английски *hardware*, то есть «металлические изделия, изделия из металла». Здесь слово *hard* означает «металлический», хотя для нас более знакомо буквальное значение «твёрдый». Так что программы – тоже в шутку – называли *software* в противоположность оборудованию *hardware*, обыгрывая буквальное значение этих слов. Получается, что «софт» – это «нечто, сделанное НЕ из металла».

Вокруг этих названий можно придумать много шуток, например название *Microsoft Mouse* можно буквально перевести как «маленькая мягкая мышка».

Для нас эта история – просто пример того, как появляются компьютерные термины и названия корпораций. Знание истории, происхождения слов помогает лучше понять значение слов и названий – и более грамотно ими пользоваться.

Запуск командного окна через [Win + R]

Командное окно можно запустить несколькими способами.

Нажимаем комбинацию клавиш [**Windows + R**].

Буква **R** – это начало слова *Run* – «Запуск программы на выполнение».

Клавиша **Windows** (или сокращённо **Win**) обычно расположена в левом нижнем ряду клавиатуры, см. рис. Она используется для вызова часто используемых функций ОС. Клавиша может отсутствовать на некоторых компьютерах.



Рис. Клавиша **Windows**

В диалоговом окне вводим название команды **cmd** и нажимаем клавишу **Enter** на клавиатуре или кнопку **OK** на экране, см рис.

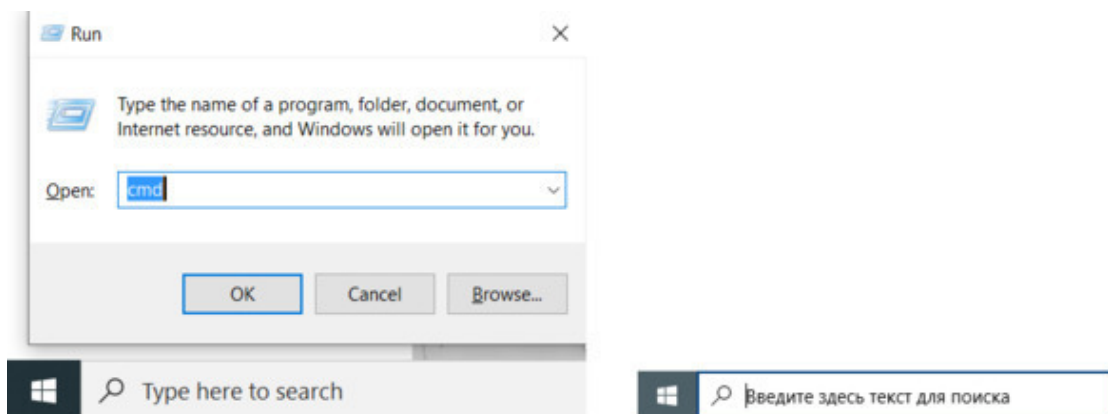


Рис. Запуск командной строки

Запускается командное окно. В правой части окна видна полоса прокрутки. На рисунке показаны основные элементы интерфейса.

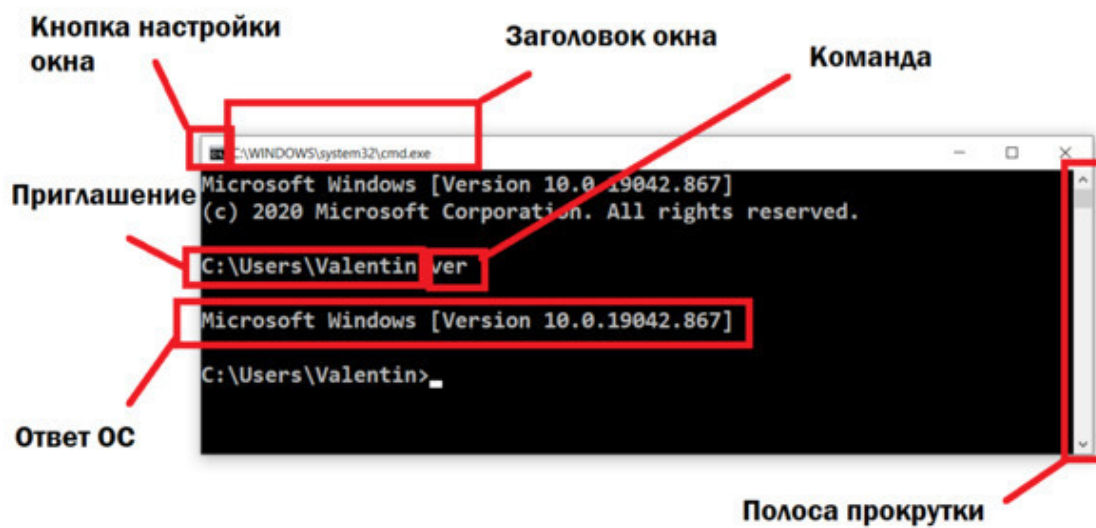


Рис. Интерфейс CMD

VER – Версия ОС

Команды можно вызывать по-разному. В простейшем случае достаточно ввести имя (название) команды и нажать **Enter**.

Задание. Введите команду **ver** и определите версию ОС.

Как видим на рис. выше, кроме номера версии, здесь можно узнать номер сборки (Build Number). Это может говорить о том, необходимо ли вам установить обновление для работы с какой-нибудь капризной программой.

Задание. Просмотрите в Википедии статью **Сборка (программирование)** и выясните, что означает слово **build** в программировании.

Кроме версии и номера сборки, возможности ОС определяются её редакцией (**Edition**).

Задание. Просмотрите в Википедии статью **List of Microsoft Windows versions** и выясните, какая редакция ОС установлена на вашем компьютере.

Команды ОС – это сокращение английских слов. В данном случае **ver** – это сокращение слова *version* – «версия».

Творческое задание. Выясните, что такое Windows NT и как это название связано с современной версией Windows.

Творческое задание. Выясните, что такое Windows Server и есть ли в нём командная строка.

История ввода команд

Командное окно запоминает, какие команды вводил пользователь.

Чтобы просмотреть историю команд, нажимаем на клавиатуре стрелки «вверх» и «вниз». Это помогает быстро повторить длинную команду. Когда предыдущая команда появилась в командной строке, её можно отредактировать – исправить ошибку или изменить. Совсем не обязательно вводить одну и ту же команду много раз вручную. Можно просто вызвать её из истории.

На полноценной клавиатуре можно обнаружить два набора стрелок – как отдельные клавиши и как клавиши цифрового блока. Чтобы использовать стрелки на числовом блоке клавиатуры, нужно нажать клавишу **NumLock**. Обычно на клавиатуре имеется индикатор нажатия этой клавиши.

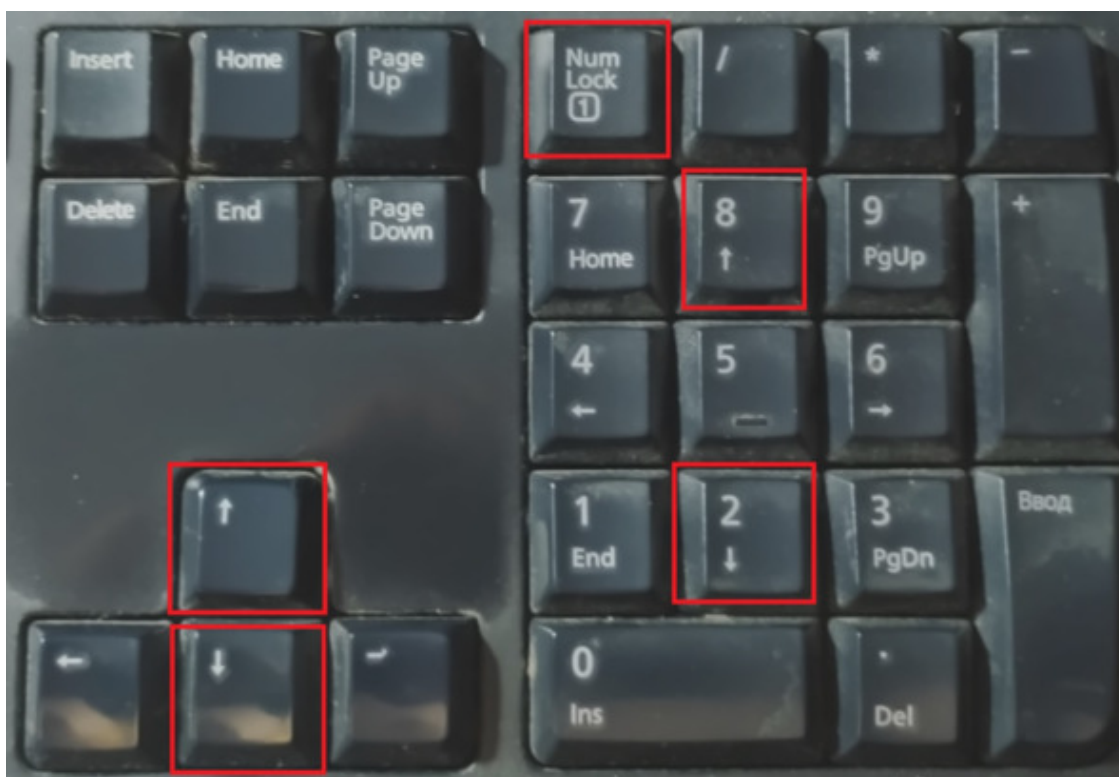


Рис. Два вида стрелок

Задание. Просмотрите в Википедии статью **NumLock** и выясните, как расшифровывается это название.

Задание. Найдите клавишу **NumLock** и ознакомьтесь с её работой. Обратите внимание на индикатор **NumLock**.

Задание. Верните предыдущую команду и нажмите **Enter**. Повторите несколько раз.

Полоса прокрутки

В правой части командного окна имеется полоса прокрутки.

Этот инструмент позволяет прокрутить окно на несколько экранов назад и ознакомиться с историей работы пользователя.

Если закрыть командное окно, история ввода команд будет потеряна.

Задание. Прокрутите командное окно вверх и вниз.

Задание. Введите команду **exit** и нажмите **Enter**.

Запуск командного окна через поиск

Второй способ запустить командное окно – воспользоваться поиском. В левом нижнем углу экрана находим окно поиска [Type here to search] и вводим название команды **cmd**, см. рис.

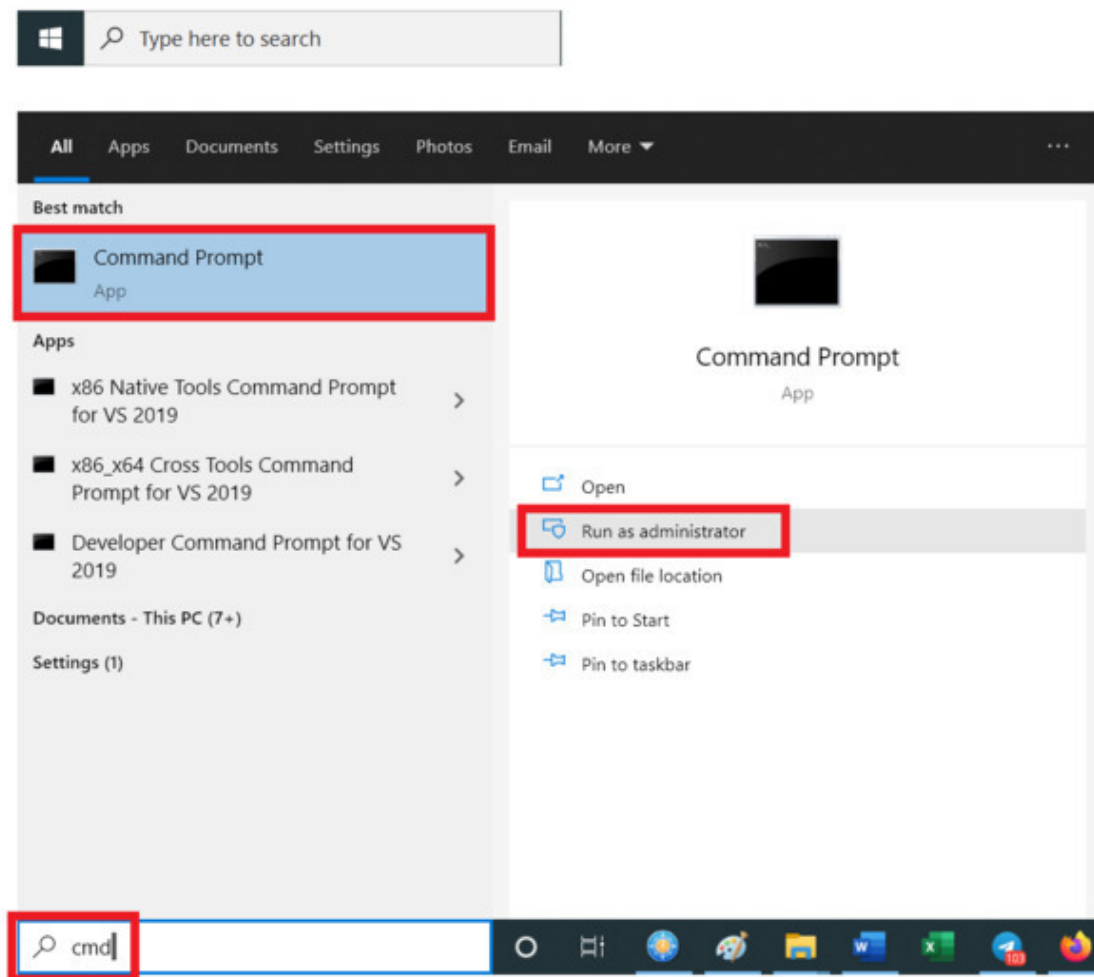


Рис. Запуск через поиск (EN)

Указав на нужное приложение и нажав правую кнопку мыши, получаем контекстное меню. Как видим, оно дублирует варианты действий, предложенные в правой части окна с результатами поиска, см. рис.

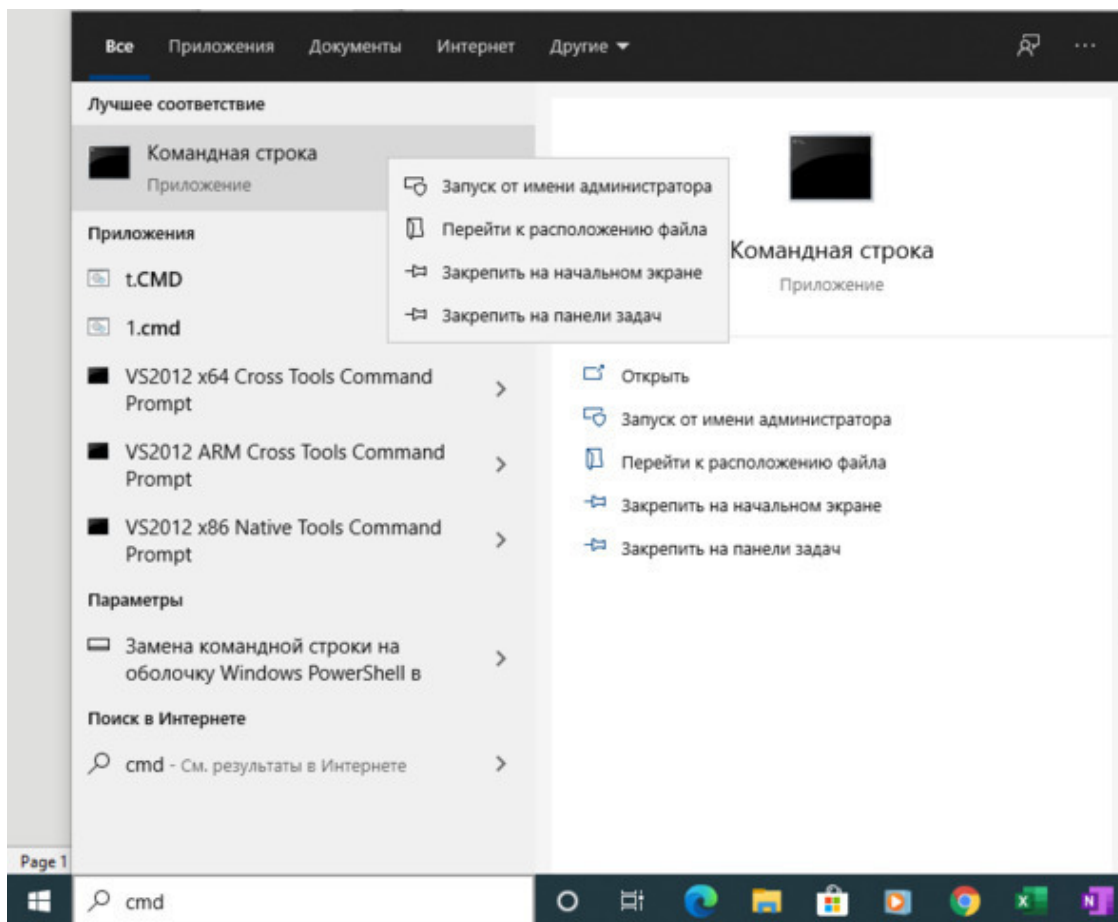


Рис. Контекстное меню (RU)

Нам сразу предлагают запустить **Command Prompt**. Буквально это выражение означает «Приглашение к вводу команд». Здесь оно означает «Командная строка» или «Командное окно». Слово **App** – это сокращение от слова Application (Software) – «Прикладная программа».

В русскоязычном варианте ОС видим название **Командная строка**.

Задание. Просмотрите в Википедии статью **Application software**. Переключитесь на русскоязычную версию статьи. Обратите внимание на термины, которыми называют программы.

В данном варианте запуска у нас появляются дополнительные возможности. Можно запустить командное окно с правами системного администратора:

Запуск от имени администратора

Run as administrator

Это может быть полезно при установке или настройке некоторых программ, когда ОС ограничивает права обычного пользователя.

Творческое задание. Просмотрите в Википедии русскоязычную статью **root** и англоязычную статью **Superuser**. Выясните, какие плюсы и минусы могут быть у доступа с правами администратора.

Язык интерфейса

Windows 10 позволяет переключить язык интерфейса без переустановки ОС.

Это может быть полезно при работе с некоторыми программами. Например, автору приходилось переключаться на русскоязычный интерфейс для заполнения налоговой декларации с помощью официального приложения ФНС. Но технологии меняются очень быстро, и сейчас можно заполнить налоговую декларацию в облаке – через диалоговое окно веб-сайта – без установки программы.

Работа с англоязычным интерфейсом помогает освоить английский язык и привыкнуть к компьютерной терминологии. Как говорят программисты, «в английском языке все слова взяты из языка C++».

Кроме того, чтение справки на английском позволяет быстрее сообразить, что означают сокращённые названия команд и их параметров. Мы увидим, что это первые буквы или начало английских слов. Так будет гораздо легче освоить работу с компьютером – все названия становятся более понятными и даже очевидными.

Чтобы переключить язык интерфейса ОС, вызываем настройки: **Start – Settings**, см. рис.

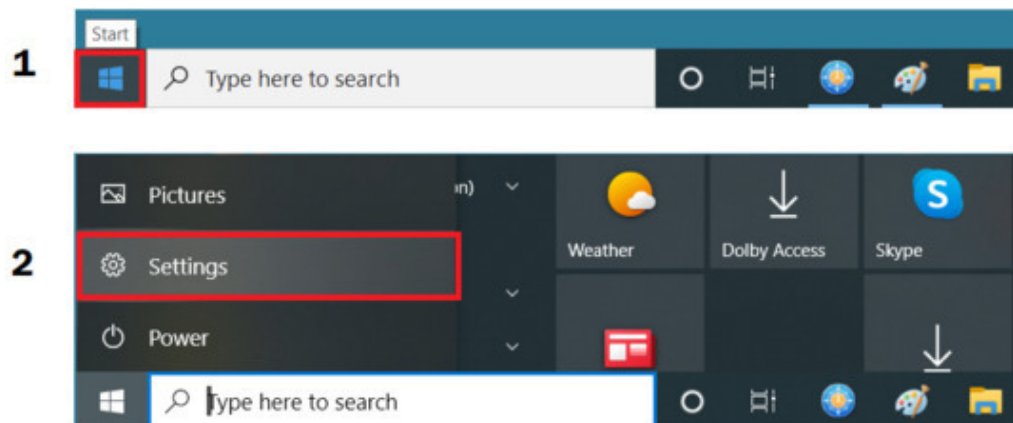


Рис. Вызов настроек ОС

Далее в меню настроек ОС выбираем настройки языка: **Time & Language**, см. рис.

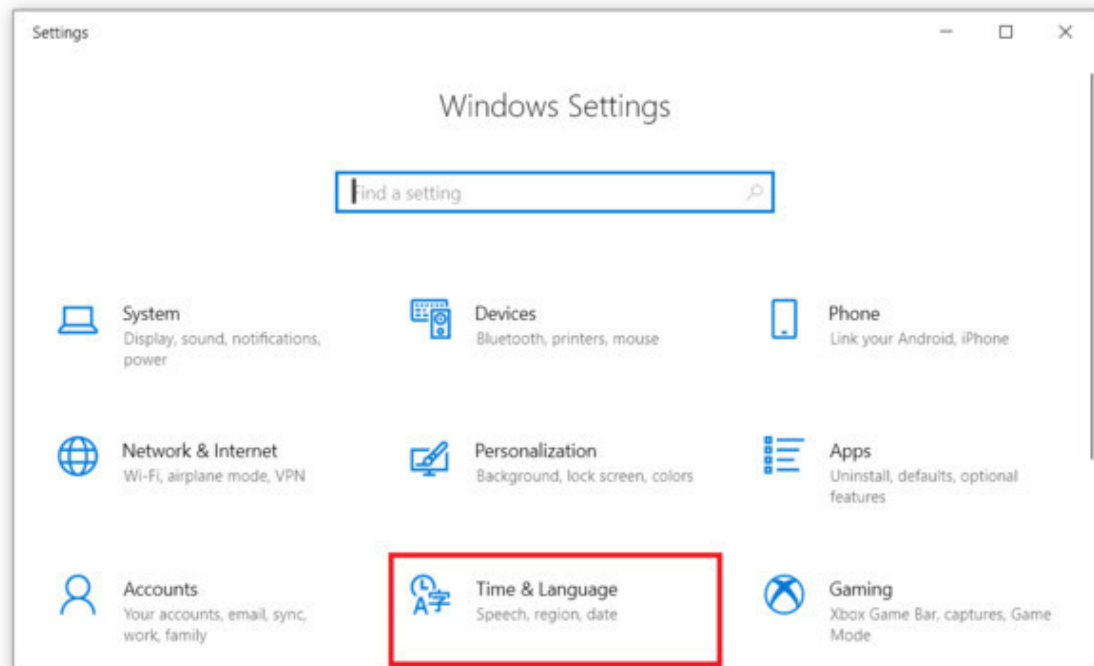


Рис. Меню настроек ОС

Переходим в раздел настройки языка **Language** и выбираем нужный язык, см. рис.

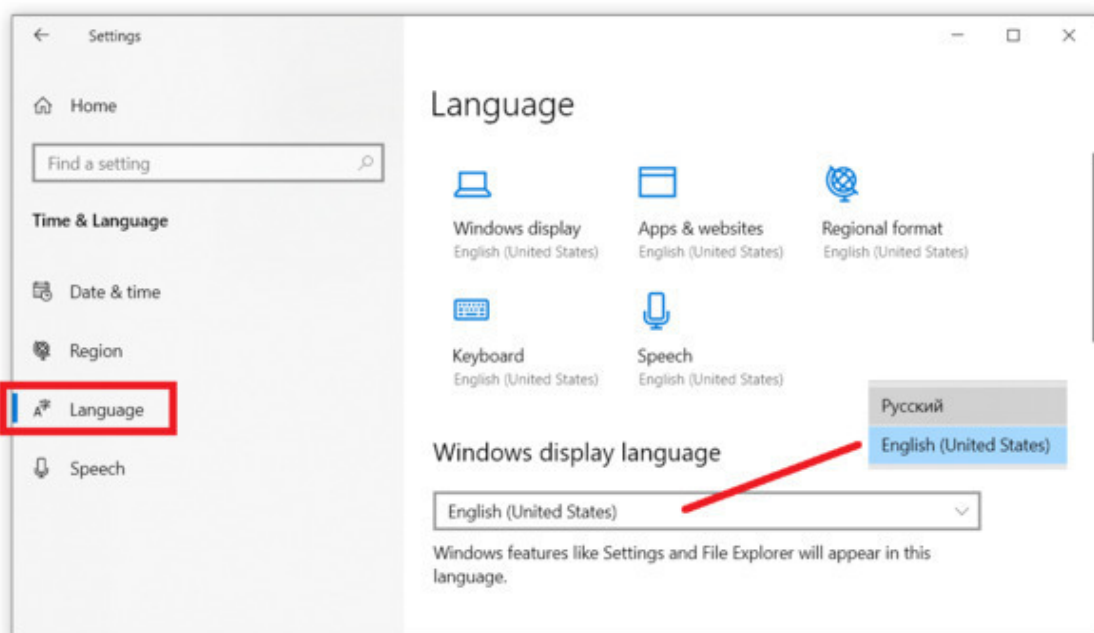


Рис. Выбор языка интерфейса ОС

После переключения языка нас предупреждают, что придётся выйти из текущей учётной записи пользователя – **Sign out**, см. рис. Другими словами, сейчас нас «выйдут» из системы. Поэтому желательно сохранить файлы, которые мы редактируем, чтобы не потерять информацию.

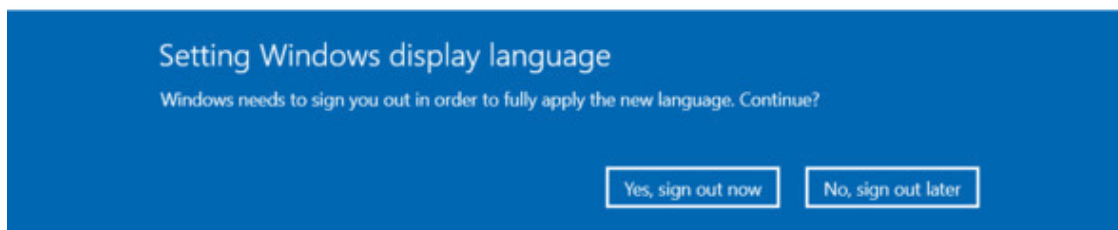


Рис. Предупреждение системы

Соглашаемся и автоматически выходим из системы.

Снова входим со своим паролем и видим новый язык интерфейса. Теперь наше командное окно (командная строка) выводит сообщения на русском языке, см. рис.

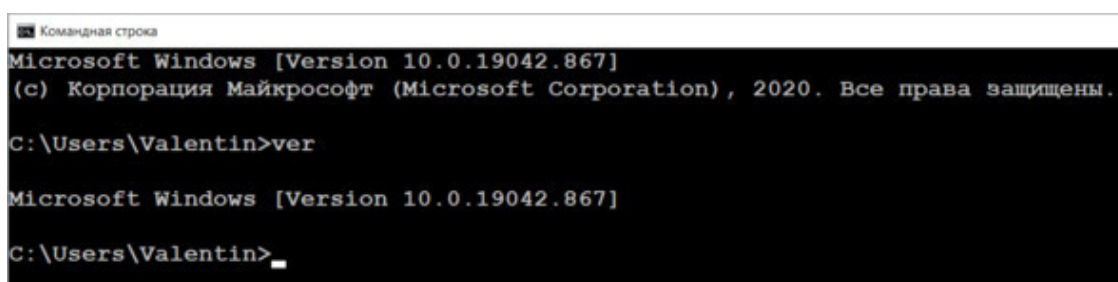


Рис. Командная строка с русскоязычным интерфейсом

Все установленные программы живы, все настройки рабочего стола сохранились, никто не пострадал. В предыдущих версиях Windows для перехода на другой язык пришлось бы заново переустанавливать всю ОС, а потом и все прикладные программы. Теперь всё стало гораздо проще и быстрее.

Осталась одна небольшая проблема. Для изменения настроек компьютера может потребоваться участие системного администратора.

Дома это легко: пользователь чаще всего сам же и является администратором.

Если же на предприятии есть всего один сисадмин на три тысячи компьютеров, то работникам можно только посочувствовать.

Компьютер может в самый неподходящий момент сообщить, что для изменения настроек ОС нужны права администратора.

Задание. Переключитесь на английский язык интерфейса. Если у вас уже настроен английский, переключитесь на русский язык, ознакомьтесь с изменениями в интерфейсе, а потом переключитесь на английский.

Запуск командного окна через меню

Есть более «официальный» способ запуска командного окна – через меню программ.

Нажимаем кнопку **Пуск**, затем прокручиваем меню программ до раздела **Windows System** и находим программу **Command Prompt**, см. рис.

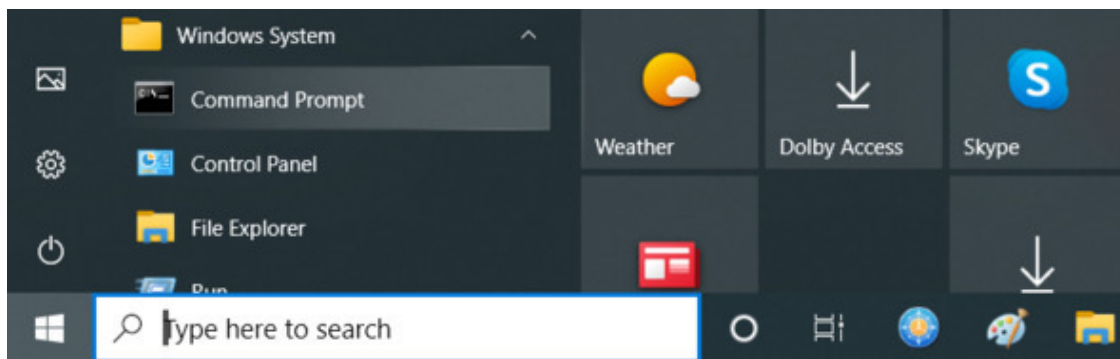


Рис. Запуск через меню

Таким образом, командное окно относится к СИСТЕМНЫМ программам. Название **Windows System** явно указывает на системные программы ОС Windows. Так что здесь слово «система» относится к операционной системе.

Задание. Запустите командное окно через меню **Пуск**.

Отметим, что в ряде случаев слово «системный» может указывать не на ОС, а на всю вычислительную систему в целом. А это уже компьютер (то есть оборудование) плюс ОС, плюс прикладные программы, плюс компьютерную сеть. И всё это хозяйство кто-то должен обслуживать и поддерживать в рабочем состоянии. Так появляется должность системного администратора (он же сисадмин). То есть администратора вычислительной (компьютерной) системы.

Задание. Просмотрите в Википедии статью **Системный администратор**, а также начало англоязычной статьи **System administrator** и уточните, что именно он обслуживает – ОС или вычислительную систему.

Каждый раз пролистывать длиннющее меню программ неудобно. Поэтому бывает полезно закрепить ссылку на нужную программу.

Открываем меню программ и находим наше командное окно. Поводим к нему курсор, нажимаем правую кнопку мыши и выбираем в контекстном меню **Pin to Start**, см. рис.

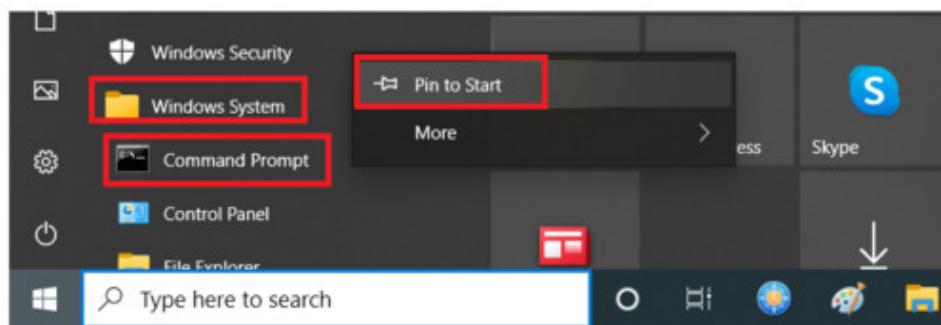


Рис. Закрепление программы в меню запуска

После этого обнаруживаем иконку командного окна в «плиточном» меню запуска программ, см. рис.

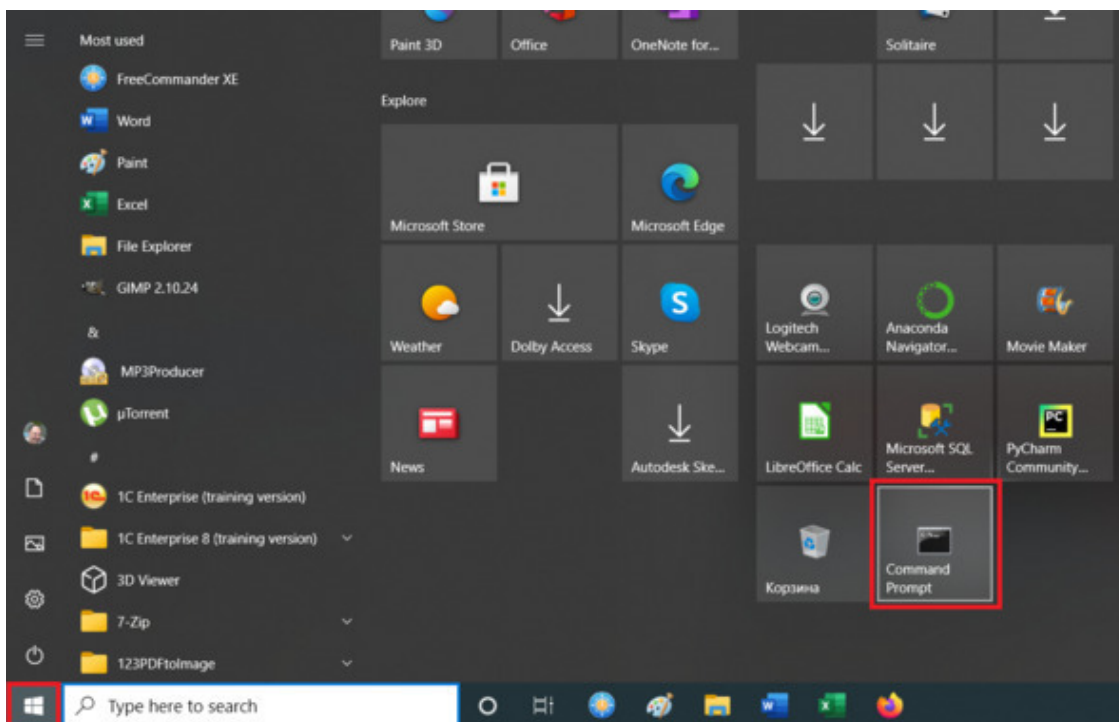


Рис. Иконка в меню запуска

К сожалению, наша иконка оказалась в самом низу списка. Можно заняться украшательством и перетащить нашу иконку наверх. Потом удалим ссылки, которые нам не так часто нужны – выбираем в контекстном меню **Unpin from Start**, см. рис.

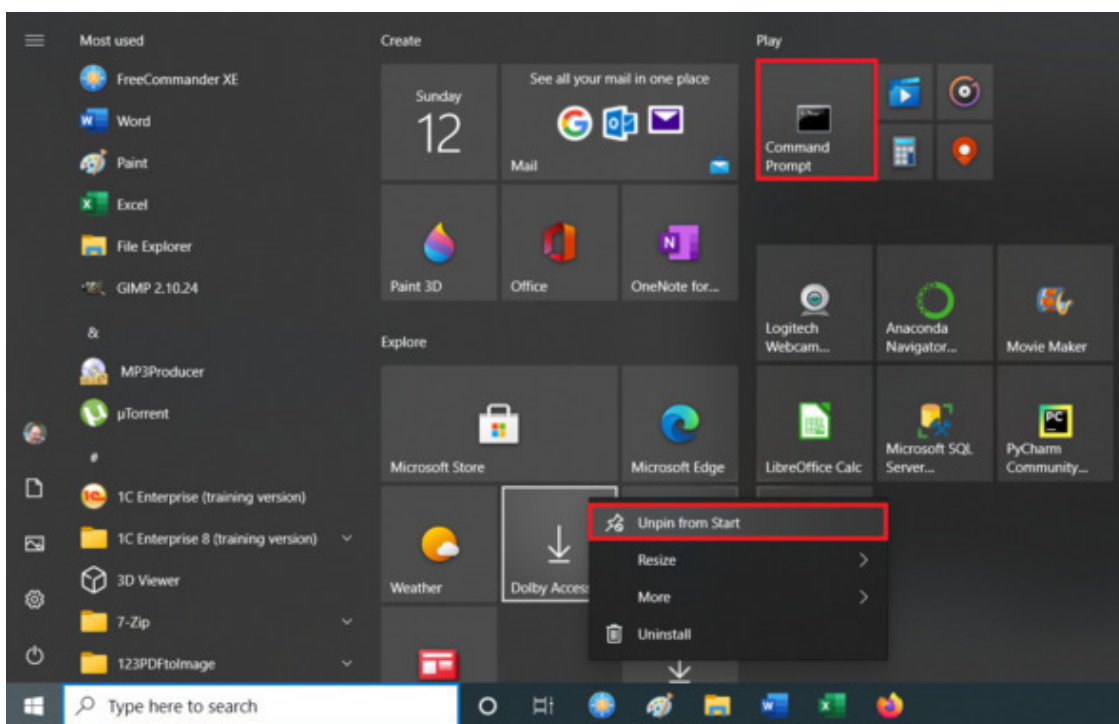


Рис. Настройка меню запуска приложений

Здесь полный простор для творчества. Иконки можно организовать в группы, сделать их разного размера или объединить в одну кнопку. Возможно, для этих настроек потребуются права администратора.

Мы можем перетаскивать иконки, переименовывать группы и менять размеры кнопок и иконками, см. рис.

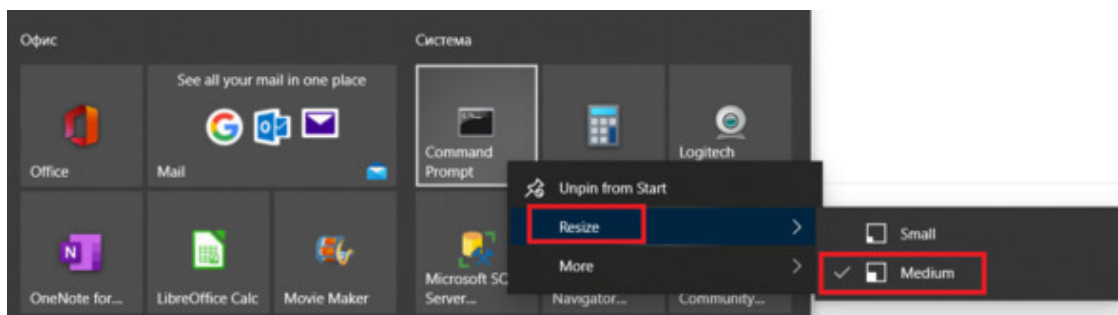


Рис. Настройка размера кнопки

Как видим, на кнопках **Pin** и **Unpin** изображена булавка для закрепления листов бумаги на доске объявлений. Булавки и пробковая доска – это во многом уже история. Сейчас чаще используют магнитики. Но идея булавки осталась. Ну а слово *Pin* буквально означает «булавка», а также «пришпилить, приколоть, закрепить с помощью булавки». В компьютерах *pin* – это просто «закрепить в меню или на экране».

Задание. Закрепите ссылку на командное окно в меню быстрого запуска приложений и выберите для неё удобное расположение.

Есть ещё одна возможность закрепить ссылку на программу. У нас на экране есть так называемая **Панель задач – Taskbar**.

Снова находим нашу иконку в меню запуска: **Start – Windows System – Command Prompt**. Вызываем правой кнопкой мыши контекстное меню и выбираем **More – Pin to taskbar**, см. рис.

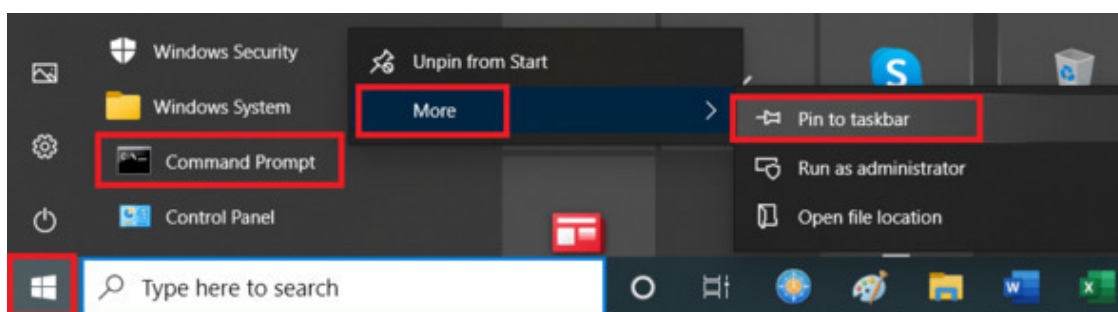


Рис. Закрепление в панели задач

Задание. Закрепите командное окно в панели задач и найдите эту иконку на экране.

Но и это ещё не всё. С помощью контекстного меню можно выяснить, где размещается программа **CMD. EXE**: **More — Open file location**. При этом открывается **Проводник – Explorer**, см. рис.

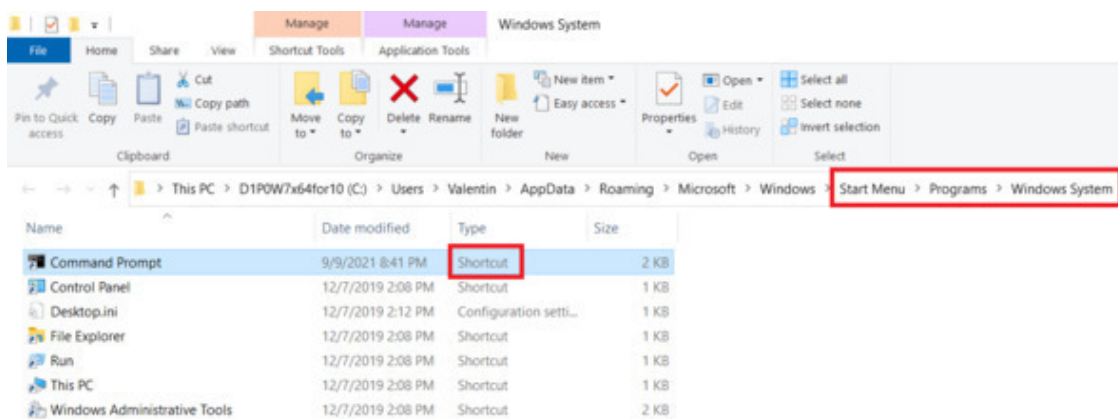


Рис. Расположение ссылки

В окне **Проводника** мы видим **Command Prompt** с указанием, что это вовсе не программа, а только **Shortcut** – ссылка, ярлык, указатель на настоящую программу.

Задание. Просмотрите в Википедии статью **Ярлык (компьютер)** и её английский вариант **Shortcut (computing)**. Выясните, что представляет собой этот самый ярлык.

Щёлкаем по ярлыку правой кнопкой мыши и выбираем в контекстном меню пункт **Properties – Свойства**, см. рис.

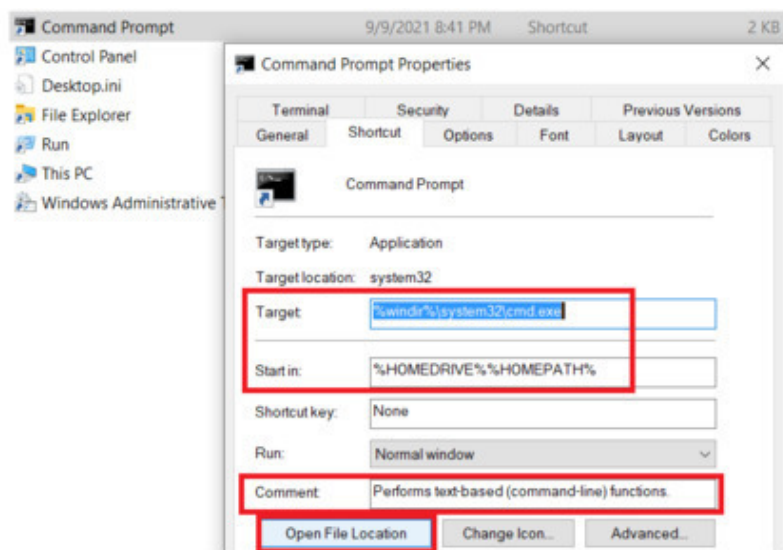


Рис. Свойства ярлыка

В окне свойств ярлыка можно увидеть, где расположена программа и какой каталог для неё будет рабочим (текущим). Можно также открыть папку с этим файлом – **Open File Location**. В новом окне **Проводника** мы видим тот самый каталог с той самой программой, см. рис.

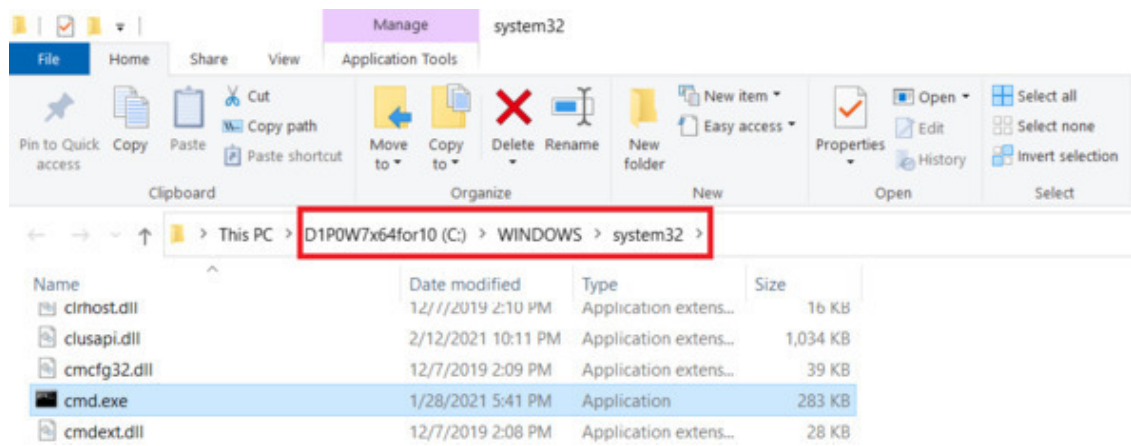


Рис. Каталог с программой

Задание. Откройте папку, в которой находится программа **CMD. EXE**.

Настройка окна

Командное окно можно настроить так, чтобы с ним было удобно работать. Вызываем меню настройки и выбираем пункт **Properties**, см. рис.

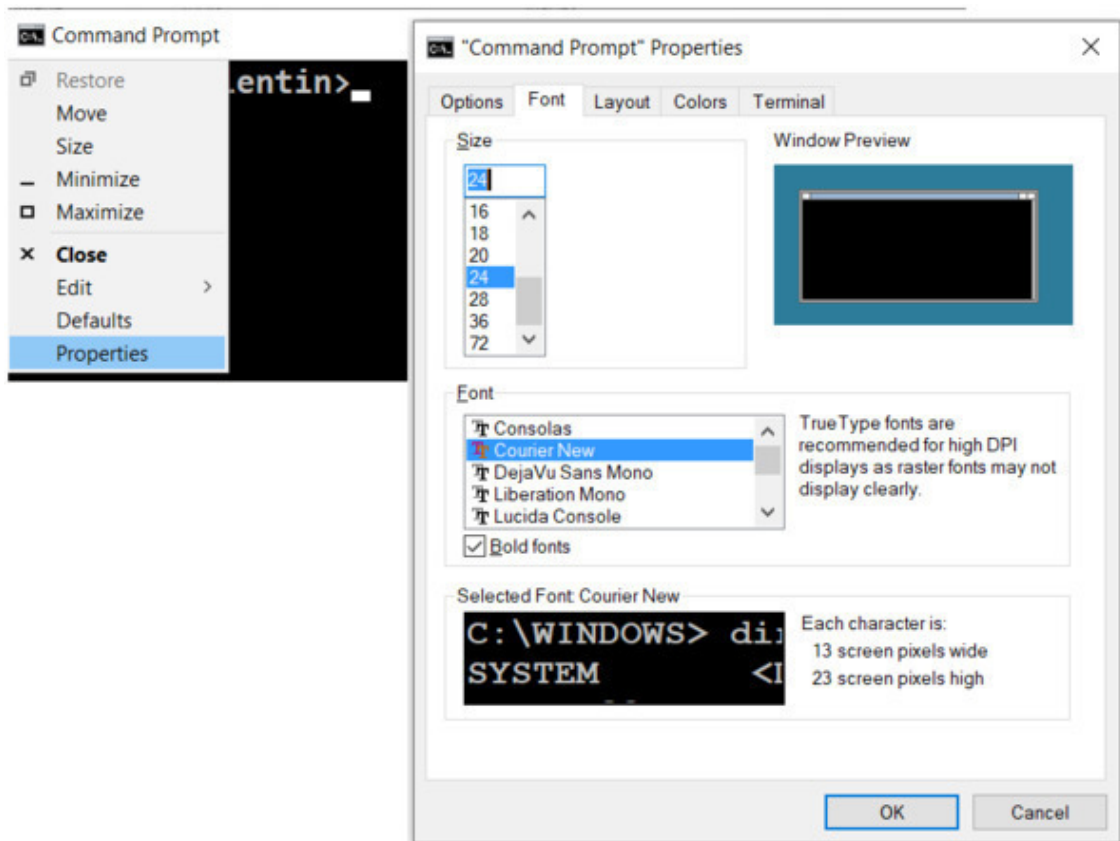


Рис. Меню командного окна (EN)

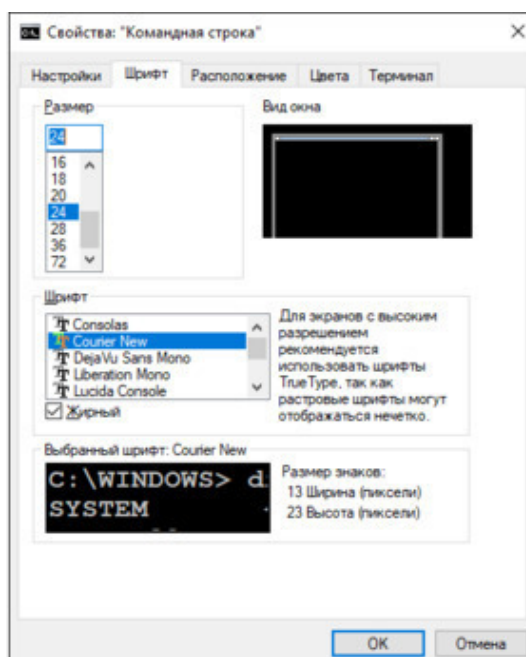


Рис. Меню командного окна (RU)

На вкладке **Font** можно настроить размер и вид шрифта.

На вкладке **Color** можно настроить цвет фона и букв.

Задание. Настройте шрифт и цвет командного окна.

Творческое задание. Выясните, в чём измеряется размер шрифта: на рисунке мы выбрали 24 чего? Как эта особая единица измерения связана привычными сантиметрами и миллиметрами?

TITLE – Заголовок окна

Командное окно оформлено так же, как и любое другое окно прикладной программы. У окна имеется заголовок.

И этот заголовок можно настроить.

Для этого используется команда **TITLE**, см. рис

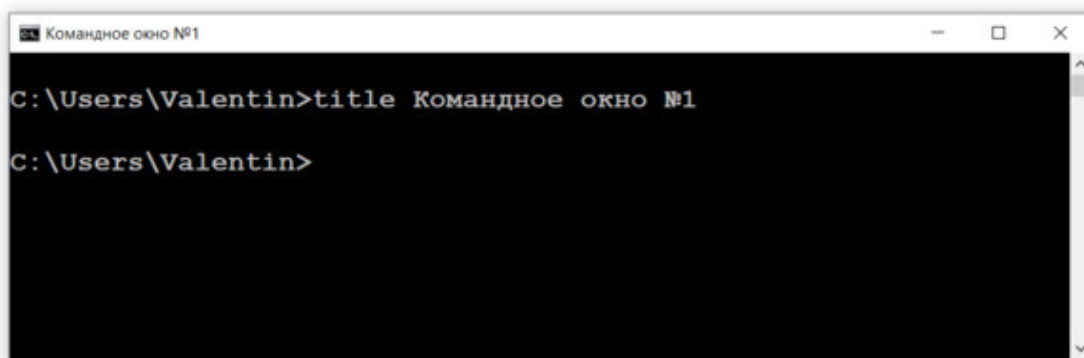


Рис. Настройка заголовка окна

В данном случае слово **TITLE** означает «заголовок».

Заголовки помогают сообщать полезную информацию, напоминать о чём-нибудь, а также различать однотипные объекты.

Обратим внимание на способ ввода команды. После названия команды через пробел указывается дополнительная информация – параметры (аргументы) командной строки.

В нашем примере через пробел указано новое название окна.

Команда **title** получает этот параметр и использует его в процессе выполнения.

Задание. Откройте три командных окна и задайте им разные заголовки.

Творческое задание. Выясните, какие ещё значения есть у английского слова **title**.

HELP – Получение справки

Для использования командной строки не обязательно запоминать все подробности и технические детали.

Достаточно представлять основные приёмы работы.

Для получения справки по конкретной команде вводим команду **help** и через пробел указываем название интересующей нас команды. Например, нас интересует команда **title**, см. рис.

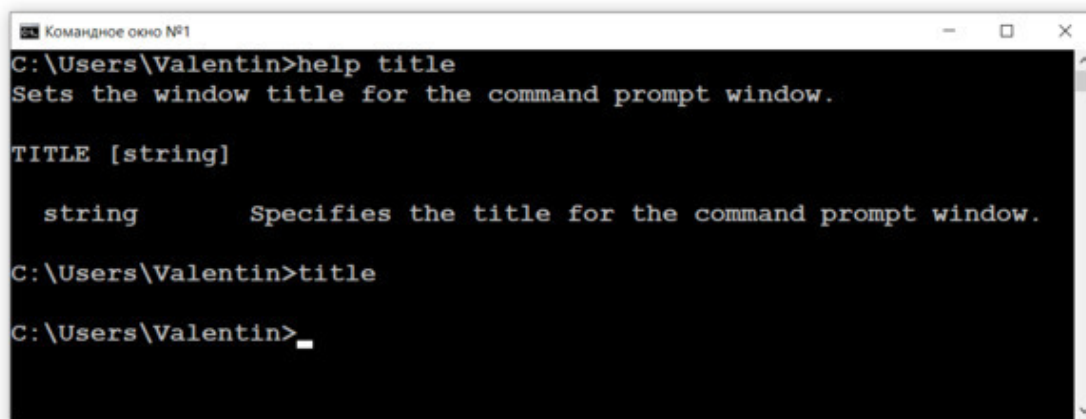


Рис. Получение справки

На экран выводится краткое описание команды.

Затем нам сообщают формат вызова команды:

TITLE [string]

В квадратных скобках указывают НЕОБЯЗАТЕЛЬНЫЕ аргументы. При вводе команды мы уже НЕ ПИШЕМ эти квадратные скобки. Посмотрим на наш пример ещё раз:

help title

Параметр в скобках – не обязательный. Команда работает и без него. Другими словами, можно вызвать команду **title** без параметров. Правда, при этом заголовок окна не изменится.

Задание. Вызовите справку для команды **title**.

Задание. Вызовите справку для команды **help**.

Задание. Вызовите команду **help** без параметров.

Судя по тексту справки, название команды **title** можно написать заглавными (большими) буквами: **TITLE**. В командном окне Windows можно использовать любой регистр при вводе команд, см. рис.

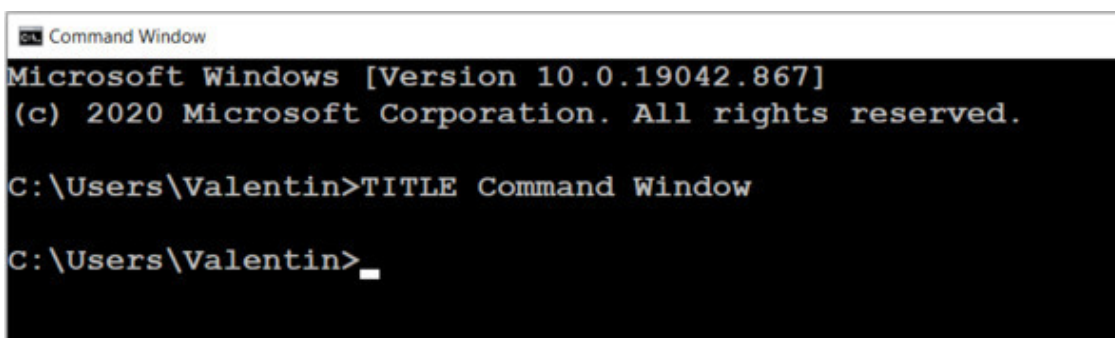
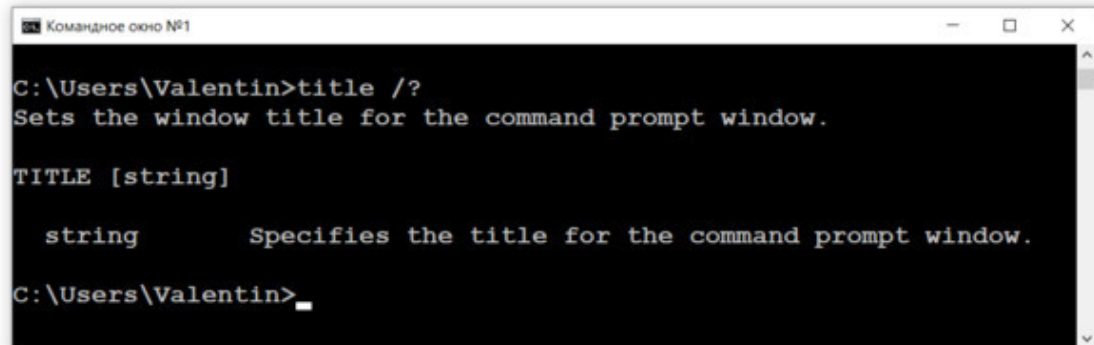


Рис. Команда заглавными буквами

Задание. Введите команду **TITLE** заглавными буквами и установите новый заголовок для окна.

Во многих случаях справку можно вызвать другим способом, см. рис.



```
Командное окно №1
C:\Users\Valentin>title /?
Sets the window title for the command prompt window.

TITLE [string]

    string          Specifies the title for the command prompt window.

C:\Users\Valentin>
```

Рис. Вызов справки

Как видим, после названия команды **title** через пробел ввели наклонную черту и вопросительный знак.

Отметим, что наклонная черта часто используется в компьютерных технологиях.

Задание. Просмотрите в Википедии статью **Косая черта**. Обратите внимание на названия этого символа и его использование в информатике.

Задание. Вызовите справку для команды **title** с помощью символа вопросительного знака.

PROMPT – Приглашение ОС

В начале командной строки ОС выводит приглашение к вводу команд.

В нашем примере это текущий каталог. Это настройка ПО УМОЛЧАНИЮ, то есть заранее заданная настройка на случай, если пользователь не использует никаких параметров или настроек. Проще говоря, если пользователь «промолчал».

При необходимости можно настроить приглашение ОС с помощью команды **prompt** – «приглашение», см. рис.



Рис. Настройка приглашения

Задание. Вызовите справку для команды **prompt**. Выясните, как называется программа для обслуживания командного окна.

Задание. Настройте приглашение так, чтобы оно содержало текущее время, пробел и текущий диск.

Многие настройки командного окна сохраняются только на время текущего сеанса работы. Есть и такие настройки, которые продолжают действовать при каждом запуске командного окна.

Задание. Закройте текущее командное окно и запустите новое. Обратите внимание на то, какие настройки сохранились, а какие – нет.

Творческое задание. Просмотрите в Википедии статью **По умолчанию** и англоязычную статью **Default (computer science)**.

START – Выполнение команды

Чтобы выполнить встроенную команду ОС или запустить файл на выполнение, достаточно ввести в командной строке название команды или файла и нажать клавишу **[Enter]** на клавиатуре.

Внешне эти два действия ничем не различаются. Мы просто печатаем несколько символов.

В работе с командной строкой мышь почти не используется. Практически все действия выполняются через клавиатуру.

Задание. Введите команду **exit** и нажмите клавишу **[Enter]**.

Задание. Посмотрите в любом словаре, что означает английское слово **exit**.

До этого момента мы работали в одном командном окне. Мы вызывали команду, и она выполнялась в том же окне.

Есть возможность запустить команду из одного окна и получить результаты в другом командном окне. Для этого служит команда **START**, см. рис.

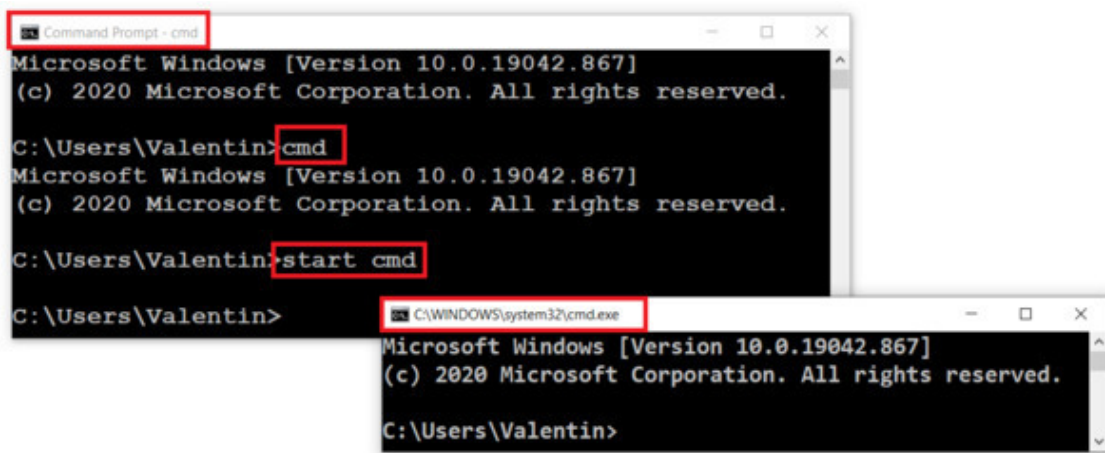


Рис. Запуск в новом окне

Здесь мы вначале запускаем команду **cmd** и получаем результат в том же окне. Затем мы пишем **start cmd** и получаем новое командное окно.

Задание. Запустите команды **help** и **start help**. Обратите внимание на появление и закрытие нового командного окна.

Задание. Запустите команды **pause** и **start pause**. Обратите внимание на появление нового командного окна.

Задание. Изучите справку для команды **pause**.

Творческое задание. Изучите справку для команды **START** и выясните, как запустить программу в «свёрнутом» окне – чтобы новое окно не мешало текущей работе и ничего не закрывало.

Запуск CMD в выбранном каталоге

В некоторых случаях требуется работать с командной строкой в определённом каталоге. В последующих разделах мы подробно разберём, как работать с каталогами.

Для начинающих есть простой обходной путь. Запускаем **Проводник – Explorer**. Переходим на нужный диск и в нужный каталог. Затем в строке адреса в верхней части окна Проводника пишем **CMD** и нажимаем **[Enter]**. Можно вводить команду и в верхнем, и в нижнем регистре. Должно сработать в любом случае.

Запускается командное окно, а в приглашении к вводу команд можно увидеть текущий каталог – тот самый, из которого мы и запустили нашу программу, см. рис.

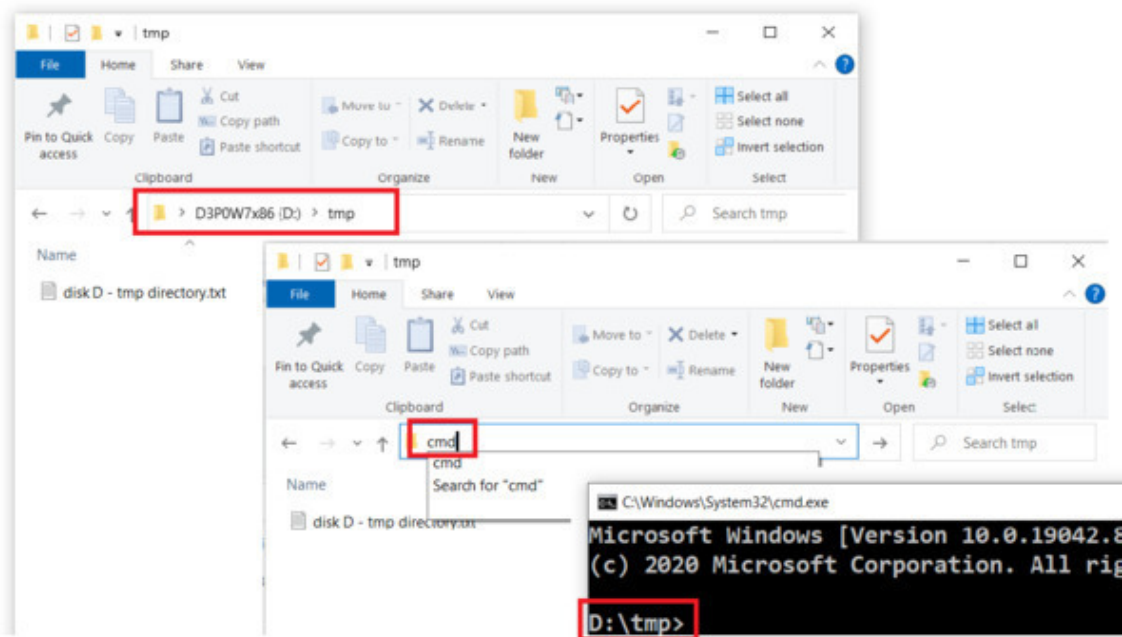


Рис. Запуск CMD из Проводника

Задание. Откройте **Проводник**, перейдите в какой-нибудь каталог и запустите **CMD**. Обратите внимание на текущий каталог в приглашении к вводу команд.

Творческое задание. Просмотрите в Википедии статью **Рабочий каталог** и английский вариант статьи – **Working directory**.

Другие способы запустить CMD

Мы рассмотрели несколько способов запуска командного окна.

Но есть и другие варианты. В интернете можно встретить подборки, в которых описано от 10 до 14 способов запустить **CMD** в Windows 10.

Задание. Найдите остальные способы запуска **CMD**, не описанные в данной работе.

Творческое задание. Выясните, как загрузить ОС Windows в режиме командной строки, и попробуйте это сделать.

Стандартный ввод-вывод

Традиционные каналы ввода и вывода информации для командной строки – это клавиатура и экран. Их называют стандартными устройствами ввода-вывода.

Слово «стандартное» здесь используется в переносном смысле. По существу, это «настройка по умолчанию», а не «соответствие требованиям документа под названием СТАНДАРТ».

Что касается слова «устройство», то здесь это некий виртуальный, программный канал, который раньше действительно был однозначно привязан к оборудованию компьютера.

Мы посмотрим, как используется этот механизм в чистом виде, а также как работает ПЕРЕНАПРАВЛЕНИЕ ввода-вывода – *Input Output Redirection*.

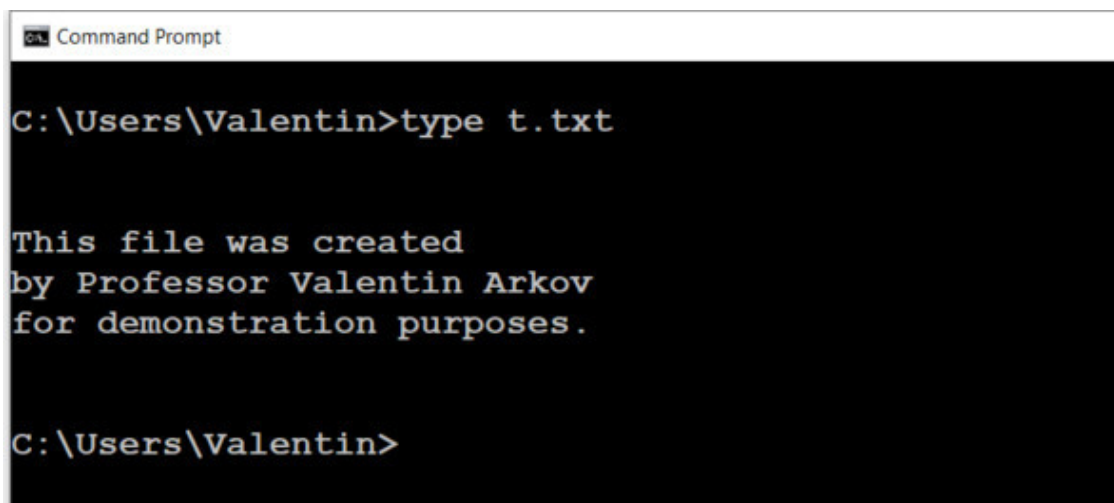
TYPE – Вывод файла на экран

Текстовый файл можно вывести на экран. В ОС Windows это можно сделать с помощью команды **type**. Это английское слово означает «напечатать». Например, на печатной машинке, а теперь и на компьютере.

Пишем команду **type** и через пробел указываем название файла, см. рис.

В нашем примере мы вывели на экран содержимое текстового файла **t.txt**:

type t.txt

A screenshot of a Windows Command Prompt window. The title bar reads "Command Prompt". The command prompt shows the path "C:\Users\Valentin>" followed by the command "type t.txt". The output of the command is displayed on the next line: "This file was created by Professor Valentin Arkov for demonstration purposes." The prompt then returns to "C:\Users\Valentin>".

```
C:\Users\Valentin>type t.txt

This file was created
by Professor Valentin Arkov
for demonstration purposes.

C:\Users\Valentin>
```

Рис. Вывод на экран

Задание. Выведите с помощью команды **type** на экран содержимое файла **help.txt** из предыдущего упражнения.

Как мы уже выяснили, название команды можно вводить в любом регистре. Убедимся, как это работает при выводе файла на экран.

Задание. Введите предыдущую команду и название файла заглавными (большими) буквами. Выясните, различает ли данная ОС большие и маленькие буквы.

Творческое задание. Изучите справку для команды **TYPE** и выясните, может ли эта команда делать что-то ещё – кроме вывода на экран.

OSK – Экранная клавиатура

Виртуальная клавиатура действует так же, как и настоящая, физическая, см. рис. **OSK** расшифровывается как On-Screen Keyboard.



Рис. Экранная клавиатура

Задание. Вызовите справку для команды **OSK**. Попробуйте объяснить результат.

Задание. Просмотрите статью **Use the On-Screen Keyboard (OSK) to type** на сайте поддержки <https://support.microsoft.com/>. Затем переключитесь на русскую версию страницы. Кнопка переключения языка находится в левом нижнем углу интернет-страницы.

Творческое задание. Поэкспериментируйте с кнопками **Options**, **Help** и **Fade** в правом нижнем углу экранной клавиатуры и выясните, что они делают.

COPY CON – Работа с консолью

В операционной системе есть названия, которые уже «заняты» и их нельзя присвоить обычным файлам. Это так называемые «зарезервированные», занятые, «забронированные» слова. Если выразиться точнее, это зарезервированные названия системных устройств ОС – Reserved device names.

Задание. Просмотрите в Википедии статью **Зарезервированное слово**.

Задание. Просмотрите в Википедии раздел **Reserved device names** англоязычной статьи **DOS**. Выясните какие названия и для каких целей зарезервированы.

Мы познакомимся с устройством под названием «консоль» – **CON**. Первоначально это было физическое устройство для связи с большим компьютером. Затем так стали называть и программы, организующие текстовый интерфейс.

Задание. Просмотрите список значений английского слова **Console** в вычислительной технике **Computing** в английской Википедии, а затем список русских значений слова **Консоль**.

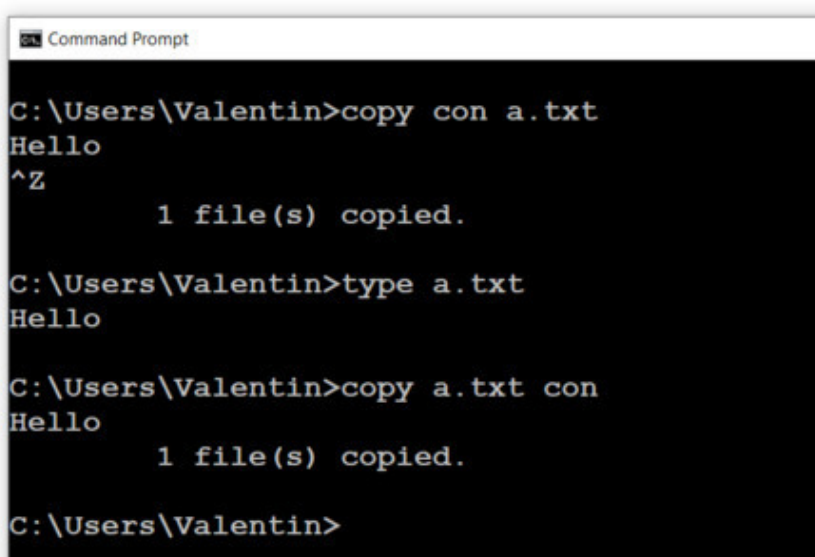
Задание. Просмотрите в Википедии англоязычную статью **System console** и обратите внимание на внешний вид физической консоли.

Обычно команда **COPY** позволяет копировать файлы. Но она также позволяет работать с консолью.

Если отправить файлы на эту «консоль», он будет напечатан на экране – как и командой **TYPE**.

Если копировать файл с консоли, можно получить текст, набираемый на клавиатуре, и записать его в файл.

При копировании файла с консоли используют команду **COPY CON** и указывают имя файла. После этого можно напечатать несколько строк на клавиатуре, а затем нажать комбинацию клавиш **[Ctrl + Z]**. На экране появятся символы **^Z**. Это сигнал об окончании ввода с клавиатуры. Введённый текст будет записан в файл, см. рис.



```
Command Prompt
C:\Users\Valentin>copy con a.txt
Hello
^Z
        1 file(s) copied.

C:\Users\Valentin>type a.txt
Hello

C:\Users\Valentin>copy a.txt con
Hello
        1 file(s) copied.

C:\Users\Valentin>
```

Рис. Копирование и консоль

Задание. Повторите описанную выше демонстрацию.

Задание. Создайте свой текстовый файл с помощью **COPY CON**. Выведите его на экран с помощью **TYPE** и **COPY CON**.

Творческое задание. Просмотрите в Википедии английскую статью **Control-Z** и выясните, что обычно обозначает такое сочетание клавиш.

COPY NUL – «Нулевое» устройство

Есть ещё одно любопытное «системное устройство» под названием **NUL**. Это устройство тоже может получать и отправлять информацию. В соответствии со своим названием это просто «ноль» – **NULL**, то есть «ничего». Устройство **NUL** принимает данные и ничего не делает. При копировании оно также может выдавать «ничего», то есть «пустой текст» – строчку текста, в котором нет ни одного символа, то есть «ноль символов».

Задание. Отправьте файл на устройство **NUL**. Скопируйте файл с устройства **NUL** и выведите его на экран. В данном задании используйте только команду **COPY**.

Творческое задание. Просмотрите в Википедии английскую статью **Null device**.

> – Перенаправление вывода

По умолчанию команды ОС выводят результаты работы на экран, то есть печатают текст на экране. Здесь экран (точнее, «текстовый» вывод на экран) – это стандартное устройство вывода **STDOUT** – используется по умолчанию.

Можно отправить выводимый текст в файл. Для этого используется символ «больше», изображающий «направление вывода» – что-то вроде «стрелки, выходящей из команды», см. рис.

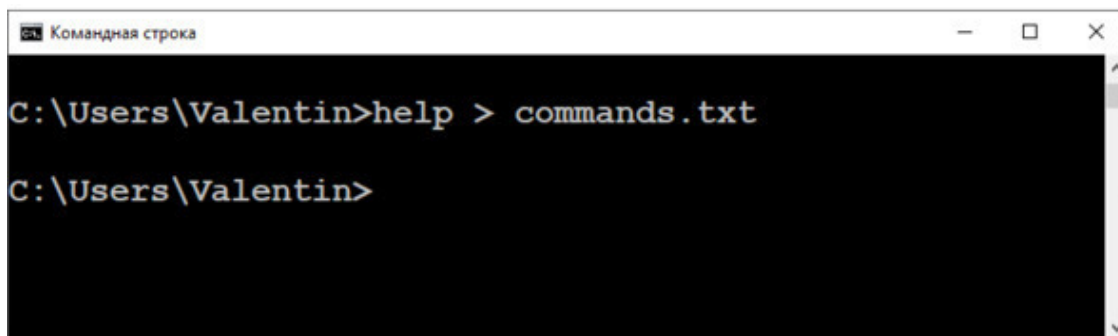


Рис. Направление вывода в файл

В данном примере мы направили вывод результатов работы команды **help** в файл под названием **commands.txt**.

Можно использовать любые команды с любыми параметрами и отправлять вывод в файл. Это файл будет создан в текущем каталоге.

Напомним, что текущий каталог указан по умолчанию в приглашении к вводу команд – в начале строки.

Задание. Вызовите команду **help help** и прочитайте справку. Затем вызовите эту команду из истории и направьте вывод в файл **help.txt**.

Задание. С помощью команды **TYPE** выведите на экран текстовый файл **help.txt**. Убедитесь, что в нём тот же текст, который мы ранее выводили на экран.

То, что мы проделали с помощью знака «больше», называется «перенаправление стандартного вывода» – *Output Redirection*. Это целая технология, о существовании которой следует знать. Может пригодиться при составлении и отладке программ, а также при работе в качестве пользователя или администратора.

Задание. Просмотрите в Википедии статью **Перенаправление ввода-вывода**, а также английский оригинал статьи **Redirection (computing)**.

>> Добавление к файлу

Отметим, что перенаправление вывода приводит к перезаписыванию выходного файла. Это означает, что мы можем потерять свои предыдущие записи.

Проведём такой опыт. Запустим две команды и перенаправим их вывод в один и тот же файл. Затем выведем содержимое файла на экран, см. рис.

```

C:\Users\Valentin>help > h.txt

C:\Users\Valentin>help help > h.txt

C:\Users\Valentin>type h.txt
Provides help information for Windows commands.

HELP [command]

    command - displays help information on that command.

C:\Users\Valentin>
  
```

Рис. Перезапись файла

Как видим, в файле записаны результаты последней команды, а всё остальное потеряно, «затёрто».

Задание. Повторите описанный эксперимент и убедитесь, что первая запись потеряна.

Чтобы не потерять текст, а просто дописывать к нему новые сообщения, используется операция добавления записи – **append**. Новый текст будет добавлен в конец файла после уже существующего текста. Для этого используют два символа «больше», см. рис.

```

C:\Users\Valentin>help help > h.txt

C:\Users\Valentin>help >> h.txt

C:\Users\Valentin>type h.txt
Provides help information for Windows commands.

HELP [command]

    command - displays help information on that command.
For more information on a specific command, type HELP command-name
ASSOC      Displays or modifies file extension associations.
ATTRIB     Displays or changes file attributes.
BREAK      Sets or clears extended CTRL+C checking.
BCDEDIT    Sets properties in boot database to control boot loading.
CACLS      Displays or modifies access control lists (ACLs) of files.
CALL       Calls one batch program from another.
CD          Displays the name of or changes the current directory.
  
```

Рис. Добавление записей

Теперь в одном файле мы имеем вывод обеих команд. На рисунке мы выделили цветом разные команды и их вывод.

Задание. Вызовите команду **help /?** несколько раз с дописыванием в конец файла. Выведите результаты на экран.

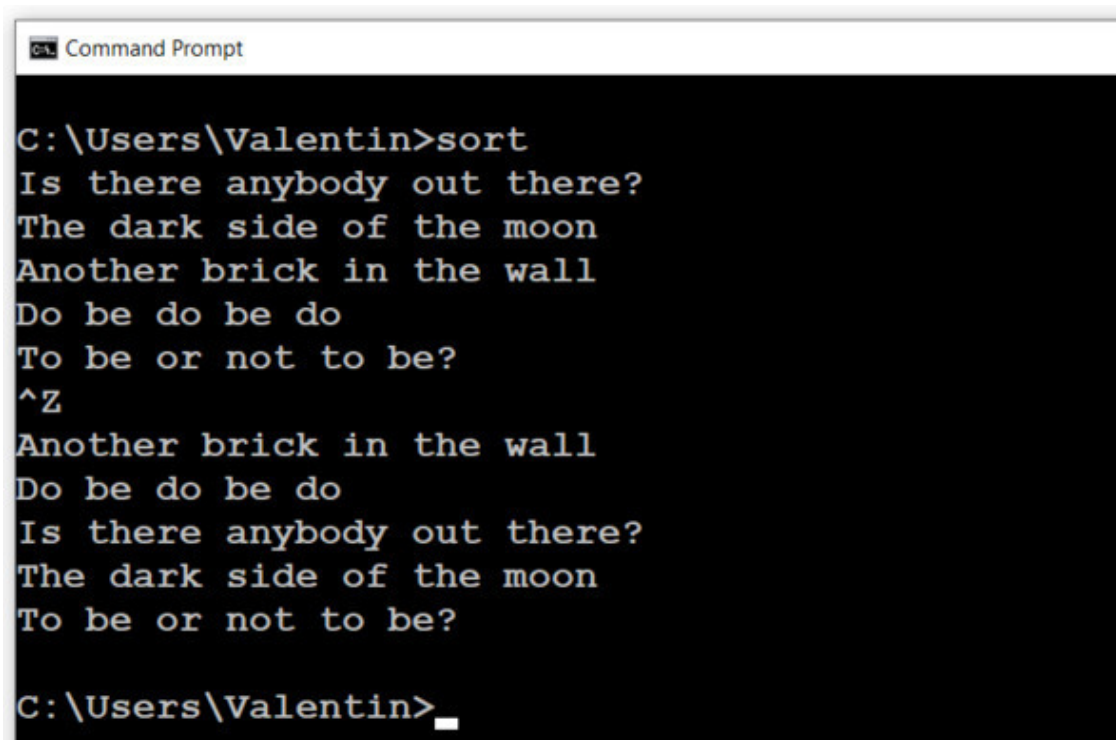
SORT – сортировка

Команда **SORT** получает несколько строк текста с клавиатуры и выводит их в нужном порядке на экран, см. рис.

После ввода каждой строки мы нажимаем клавишу **[Enter]**.

После ввода последней строки нажимаем комбинацию клавиш **[Strl + Z]**, а затем **[Enter]**.

Что-то знакомое – по предыдущим упражнениям.



```
Command Prompt

C:\Users\Valentin>sort
Is there anybody out there?
The dark side of the moon
Another brick in the wall
Do be do be do
To be or not to be?
^Z
Another brick in the wall
Do be do be do
Is there anybody out there?
The dark side of the moon
To be or not to be?

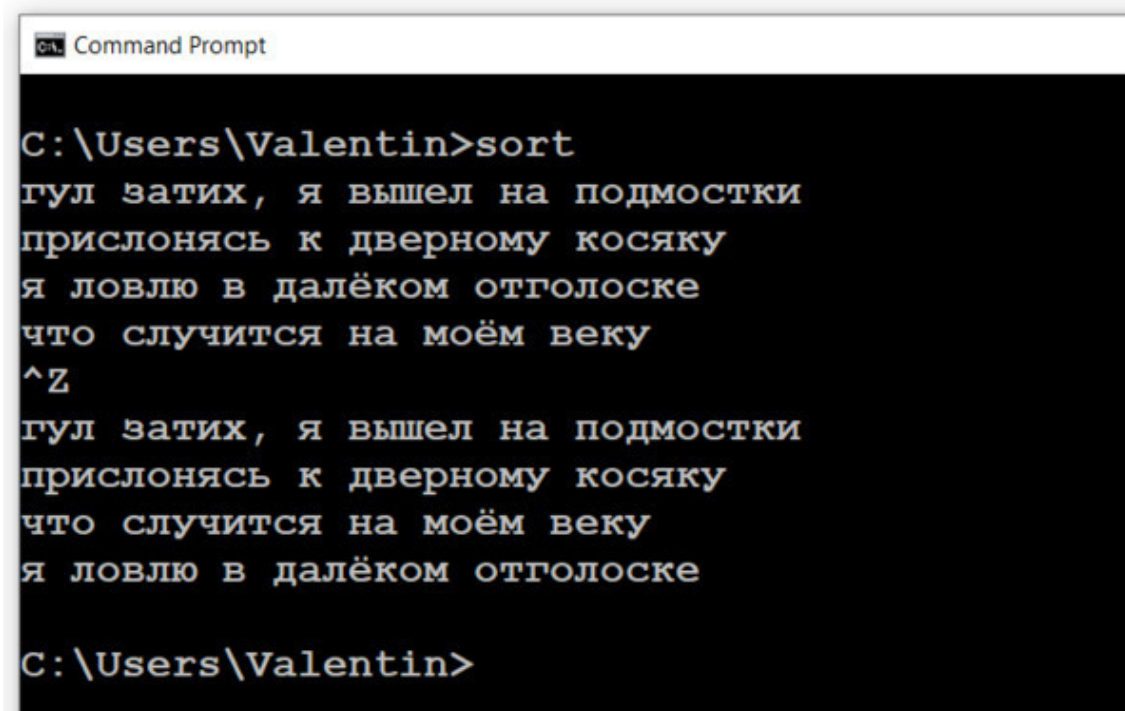
C:\Users\Valentin>_
```

Рис. Сортировка строк

Как видим, результаты сортировки выводятся на экран – в алфавитном порядке.

Задание. Вызовите команду **SORT** и введите с клавиатуры несколько строк.

Проверим, как происходит сортировка русского текста, см. рис.



```
Command Prompt

C:\Users\Valentin>sort
гул затих, я вышел на подмости
прислонясь к дверному косяку
я ловлю в далёком отголоске
что случится на моём веку
^Z
гул затих, я вышел на подмости
прислонясь к дверному косяку
что случится на моём веку
я ловлю в далёком отголоске

C:\Users\Valentin>
```

Рис. Сортировка строк

Задание. Проверьте, как работает сортировка, если разные строки начинаются с русских и английских букв, а также с цифр.

Задание. Изучите справку для команды **SORT**. Обратите внимание на следующие ключи:

/R

/O

Задание. Проверьте, как работает сортировка с ключами **/R** и **/O**.

<– Перенаправление ввода

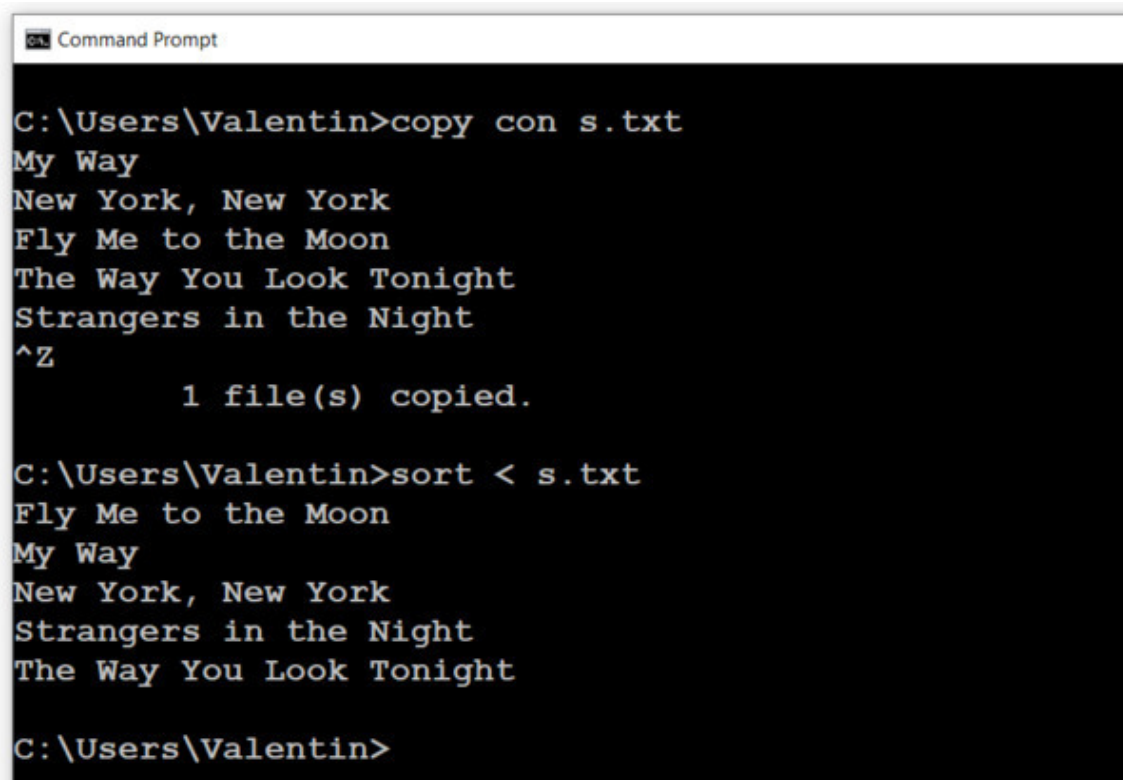
По умолчанию для ввода текста в командной строке используется клавиатура. Это стандартное устройство ввода **STDIN**.

Если какая-нибудь команда получает данные с клавиатуры, этот ввод можно перенаправить. Можно взять данные из файла, а программа будет с ними работать – как будто мы напечатали этот текст на клавиатуре. При этом сама программа не изменилась. Изменился только способ вызова команды.

Для перенаправления ввода используют символ «меньше». Это напоминает стрелку влево. То есть это ввод для команды, которая находится слева от знака «меньше»:

команда <файл

Рассмотрим перенаправление ввода на примере сортировки. Создадим текстовый файл с помощью команды **COPY CON**. Затем отправим файл на вход команды **SORT**, см. рис.



```
Command Prompt

C:\Users\Valentin>copy con s.txt
My Way
New York, New York
Fly Me to the Moon
The Way You Look Tonight
Strangers in the Night
^Z
        1 file(s) copied.

C:\Users\Valentin>sort < s.txt
Fly Me to the Moon
My Way
New York, New York
Strangers in the Night
The Way You Look Tonight

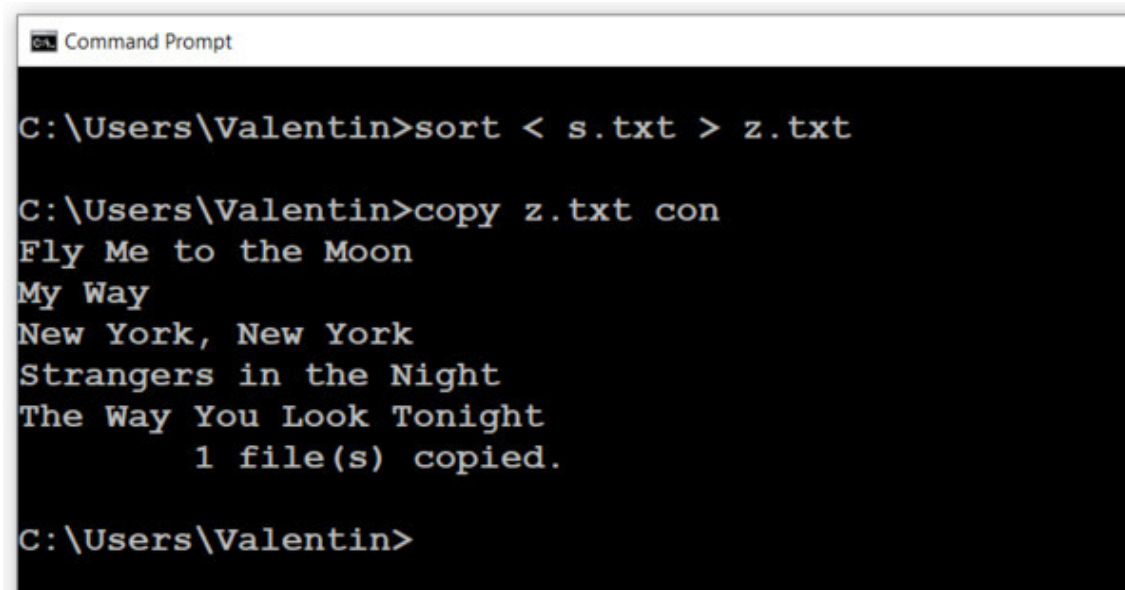
C:\Users\Valentin>
```

Рис. Перенаправление ввода

Задание. Создайте текстовый файл и проведите его сортировку с выводом на экран.

Мы рассмотрели перенаправление ввода и перенаправление вывода. Эти два действия можно скомбинировать в одной команде, см. рис.

Здесь мы получаем текст из файла, сортируем его в алфавитном порядке и отправляем результаты сортировки в другой файл.



```
Command Prompt

C:\Users\Valentin>sort < s.txt > z.txt

C:\Users\Valentin>copy z.txt con
Fly Me to the Moon
My Way
New York, New York
Strangers in the Night
The Way You Look Tonight
1 file(s) copied.

C:\Users\Valentin>
```

Рис. Перенаправление ввода и вывода

Задание. Проведите сортировку текстовых строк файла и отправьте результаты в другой файл. Выведите содержимое отсортированной файла на экран командой **TYPE**.

Творческое задание. Изучите справку для команды **SORT** и выясните, для чего используется опция **/+n**. Затем проверьте, как это работает на вашем компьютере.

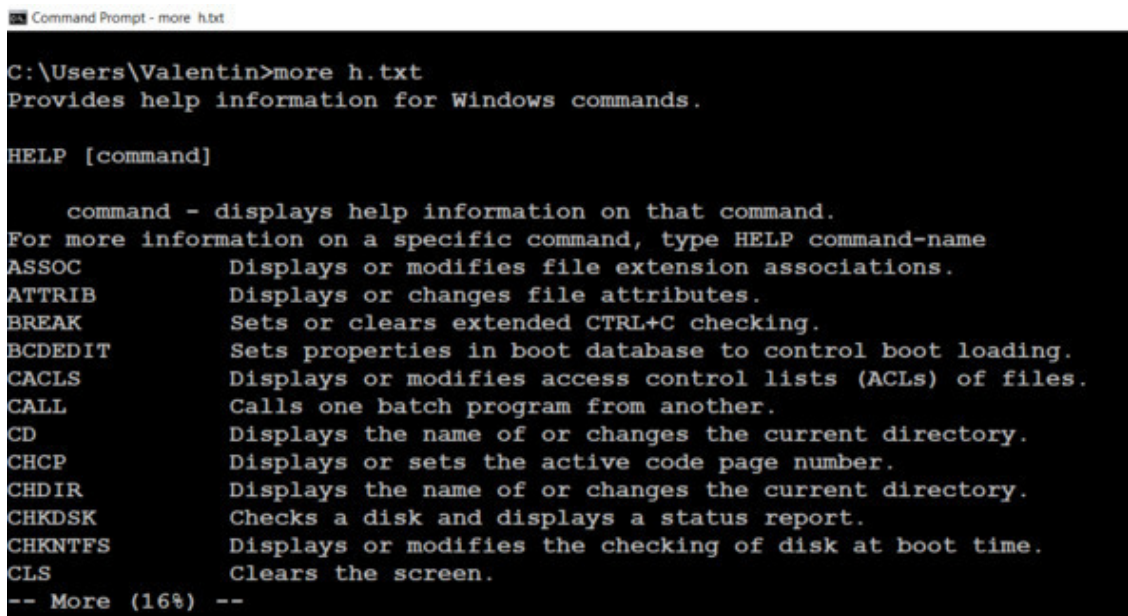
MORE – Постраничный вывод

При выводе текста на экран возникает одна проблема. Длинный текст выводится полностью, от начала до конца. В результате мы видим мелькание букв, а затем последние несколько строк и приглашение к вводу следующей команды.

Для чтения всего текста придётся прокрутить экран назад, а это не всегда удобно.

В командной строке есть полезная команда **MORE** для вывода текста небольшими порциями, по страницам. Буквальное значение английского слова MORE – «ещё». Здесь мы просим вывести «ещё одну страницу», «следующую порцию» текста на экран.

Самый простой вариант использования данной команды – это вывод файла на экран. Мы вводим команду **MORE** и указываем название файла, см. рис.



```

Command Prompt - more h.txt

C:\Users\Valentin>more h.txt
Provides help information for Windows commands.

HELP [command]

    command - displays help information on that command.
For more information on a specific command, type HELP command-name
ASSOC          Displays or modifies file extension associations.
ATTRIB         Displays or changes file attributes.
BREAK          Sets or clears extended CTRL+C checking.
BCDEDIT        Sets properties in boot database to control boot loading.
CACLS          Displays or modifies access control lists (ACLs) of files.
CALL           Calls one batch program from another.
CD             Displays the name of or changes the current directory.
CHCP           Displays or sets the active code page number.
CHDIR          Displays the name of or changes the current directory.
CHKDSK         Checks a disk and displays a status report.
CHKNTFS        Displays or modifies the checking of disk at boot time.
CLS            Clears the screen.
-- More (16%) --

```

Рис. Постраничный вывод файла

Как видим, перед нами первая «страница» текста, которая умещается на один экран. В нижней строке нам намекают, что это работа команды **MORE** и сообщают, сколько процентов файла уже было напечатано на экране.

Чтобы вывести следующую страницу текста, нажимаем клавишу «Пробел» – [Space].

Задание. Используя команду **MORE**, выведите свой длинный файл из предыдущих упражнений на экран.

Есть вторая важная клавиша для управления постраничным выводом. Это прерывание вывода. Текст может быть очень длинным, и нам не всегда нужно читать всё подряд. Эту секретную клавишу можно найти в описании команды **MORE**.

Задание. Изучите справку для команды **MORE** и выясните, как выйти из режима постраничного вывода – не добираясь до последней страницы. Снова запустите постраничный вывод длинного файла и выйдите из этого режима.

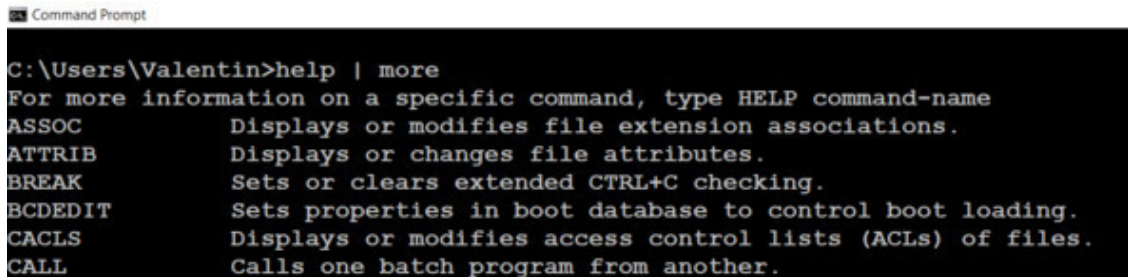
Творческое задание. Изучите справку для команды **MORE** и выясните, какие команды доступны в режиме постраничного просмотра файла.

| – Конвейер

Операционная система позволяет объединить несколько команд в конвейер. Соответствующее английское название – *Pipe*. Здесь команды будут выполняться одна за другой. При этом вывод одной команды направляется на вход следующей команды.

Для организации конвейера в командной строке используют символ «вертикальная черта» |.

Мы рассмотрим простой конвейер с использованием команды **MORE**, см. рис.



```

C:\Users\Valentin>help | more
For more information on a specific command, type HELP command-name
ASSOC          Displays or modifies file extension associations.
ATTRIB         Displays or changes file attributes.
BREAK          Sets or clears extended CTRL+C checking.
BCDEDIT        Sets properties in boot database to control boot loading.
CACLS          Displays or modifies access control lists (ACLs) of files.
CALL           Calls one batch program from another.
  
```

Рис. Конвейер

В данном примере команда **HELP** обычно выводит список команд на экран. Но мы перенаправляем этот поток стандартного вывода на вход следующей команды. Теперь команды **MORE** организует нам постраничный вывод информации на экран. Как перейти на следующую страницу и выйти из такого режима, мы уже знаем.

Задание. Повторите описанный опыт. Пролистайте пару страниц, затем выйдите из режима постраничного просмотра.

Мы можем соединить любое количество команд в длинный конвейер. Нужно просто разделять команды символом вертикальной черты.

Задание. Введите следующую команду:

HELP | SORT | MORE

Объясните полученные результаты.

Для лучшего понимания работы конвейера неплохо было бы что-нибудь почитать – дополнительно к чтению методички или учебного пособия.

Задание. Просмотрите в Википедии статью **Вертикальная черта** и её английский оригинал **Vertical bar**.

Творческое задание. Просмотрите в Википедии статьи **Конвейер (Unix)**, **Pipeline (Unix)**, **Pipeline (software)**. Выясните, на что намекает английское слово *Pipeline* в названии вычислительного конвейера.

& – Последовательный запуск команд

В командной строке есть много других инструментов для управления командами.

Символ «амперсанд» **&** позволяет запускать команды – одну за другой. При этом ввод и вывод никак не затрагивается.

Задание. Запустите показанные ниже команды и обратите внимание на результаты их выполнения:

```
HELP /? | START CMD  
HELP /? & START CMD  
HELP /? & START «NEW CMD» CMD
```

Задание. Просмотрите статью **Амперсанд** и английский оригинал статьи **Ampersand**. Выясните, что означает название этого символа.

Творческое задание. Выясните, как используют сочетания символов **||** и **&&** при работе в командной строке.

stdio. h – Стандартный ввод-вывод в Си

Мы рассмотрели поведение команд ОС и работу с вводом-выводом. Такие же приёмы можно использовать с любой программой, которую мы запускаем в командной строке.

Прикладная программа, которая работает в командной строке, называется «консольное приложение». Напомним, что «консоль» – это другое название для командной строки, а «приложение» – это то же самое, что и прикладная программа.

Далее мы создадим простейшие программы, которые что-нибудь выводят на экран и что-нибудь читают с клавиатуры. Затем мы увидим, как на этих программах сработает перенаправление ввода-вывода и организация конвейера.

MinGW – Среда разработки

Нас интересует среда разработки MinGW, которая фактически реализует компилятор GCC языка программирования Си. При этом мы будем работать в переносимой версии среды MinGW, которая не требует установки в операционную систему.

Задание. Просмотрите в Википедии следующие статьи:

- **MinGW;**
- **Mingw-w64;**
- **GNU Compiler Collection;**
- **Portable application;**
- **Переносимое приложение.**

Процедура загрузки и настройки компилятора показана, например, в ролике:

Install MinGW-W64 from archive on Windows

<https://youtu.be/1TPmiiAqmlc>

Скачиваем архив в формате **ZIP** по адресу:

<https://sourceforge.net/projects/mingw-w64/>

Этот адрес работает на момент написания данного учебного пособия. Если данный проект будет закрыт, можно поискать любую переносимую версию MinGW с компиляторами GCC.

Переходим на вкладку **Files** и скачиваем архив **x86_64-posix-seh**. Далее распаковываем архив в удобном месте на рабочем диске.

Находим в **Проводнике** каталог **bin** и копируем путь к каталогу с исполняемыми файлами в буфер обмена, см. рис.

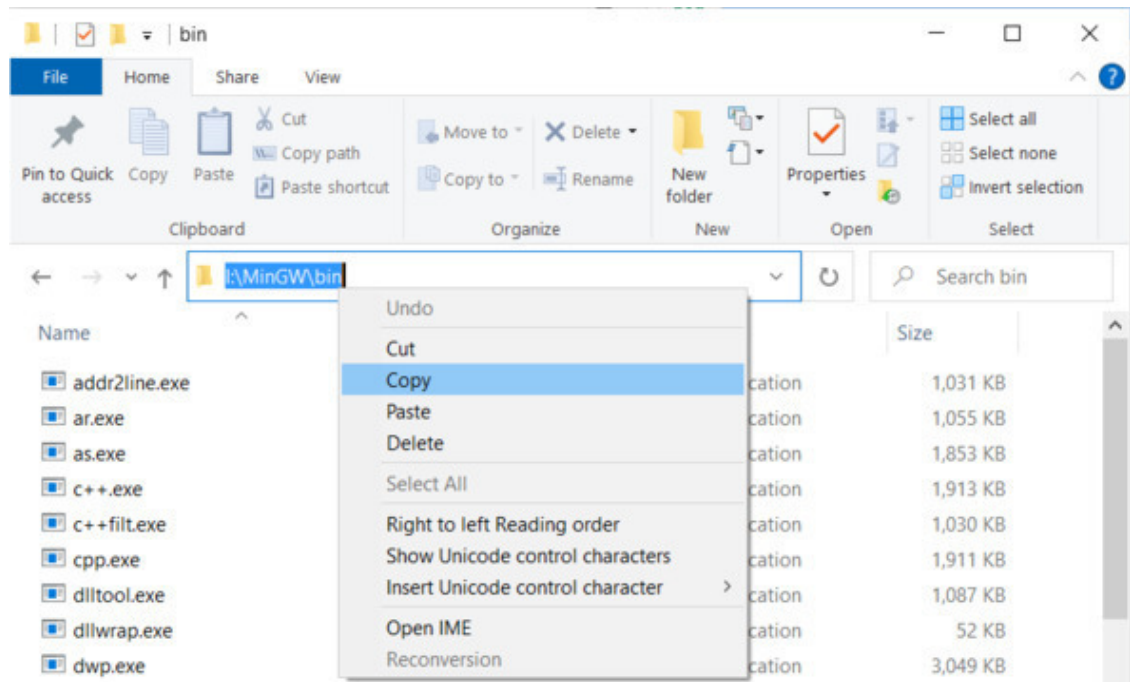


Рис. Путь к каталогу

Далее щёлкаем правой кнопкой по кнопке **Start** и выбираем пункт меню **System**.

В строке поиска вводим **environment** и выбираем настройку для своей учётной записи – **Edit environment variables for your account**, см. рис.

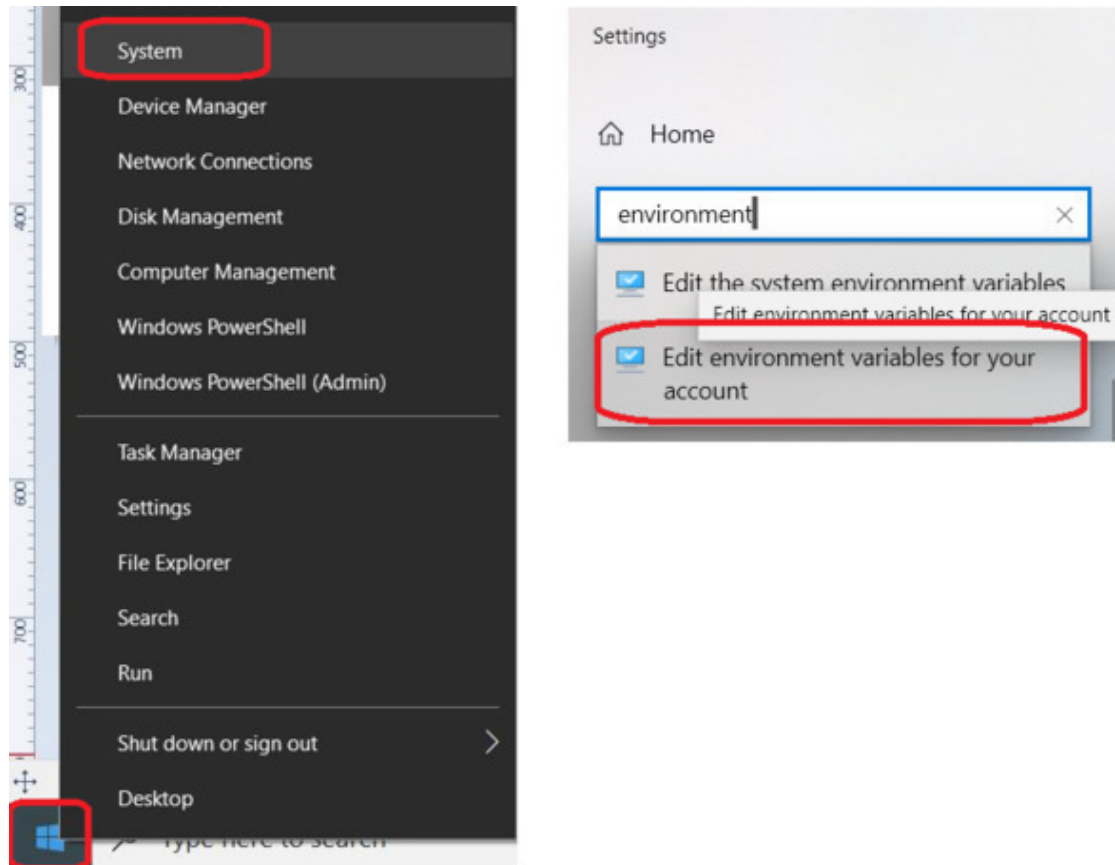


Рис. Вызов редактирования параметров окружения

Далее выбираем настройки параметров окружения для текущего пользователя – **User variables for...**, указываем переменную окружения **Path** и нажимаем кнопку **Edit**.

В диалоговом окне **Edit environment variable** нажимаем кнопку **New** и вставляем из буфера путь к исполняемым файлам компилятора, см. рис.

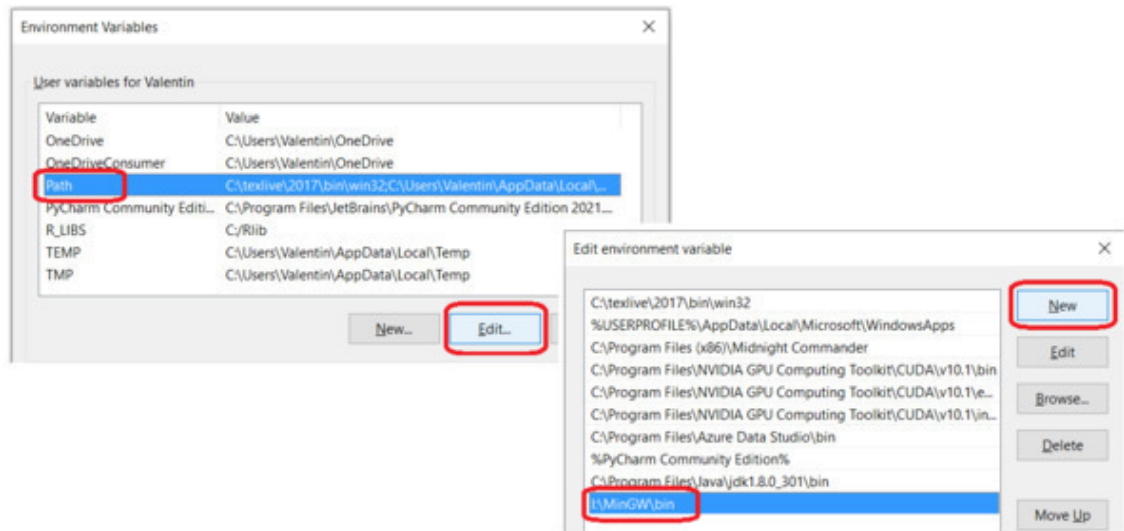


Рис. Настройка параметра **Path**

Нажимаем **ОК** и последовательно закрываем все окна настройки ОС.

Если пользователь не имеет доступа к настройкам ОС, можно установить переменную окружения **Path** обычным путём с помощью команды **set Path**, как мы уже проделали в предыдущих заданиях.

Программа с выводом на экран

Составим программу, которая выводит сообщение на экран – устройство стандартного вывода.

Создаём рабочий каталог, запускаем командную строку в этом каталоге и создаём первую программу **Hello World**.

Для работы с текстом программы можно использовать **Блокнот**, см. рис.

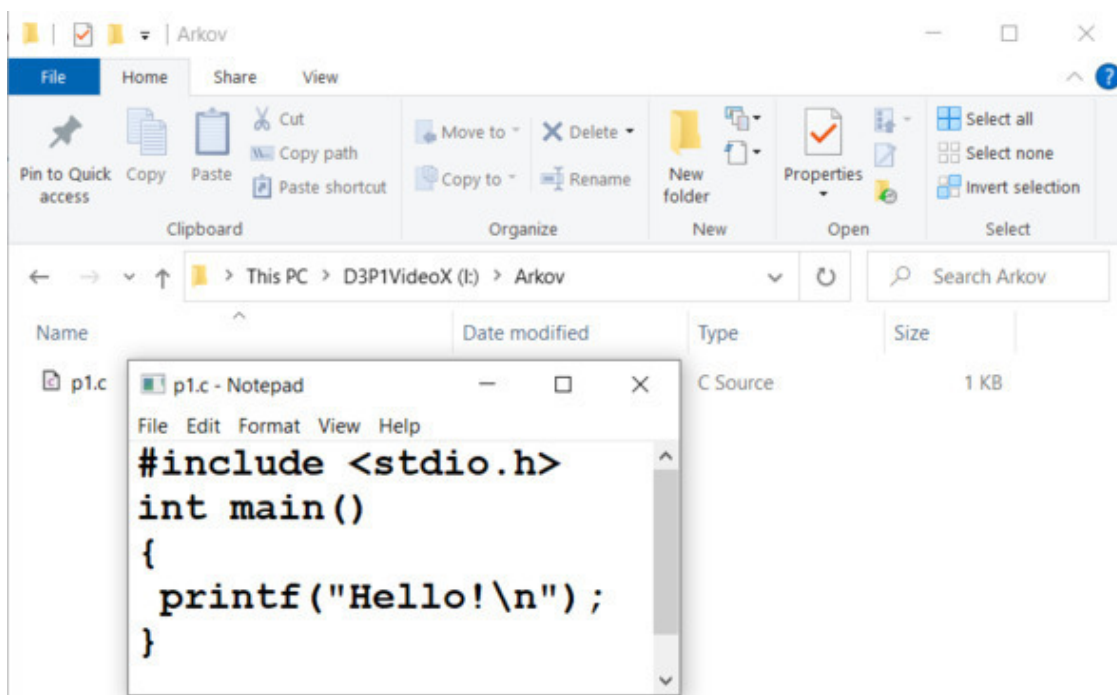


Рис. Первая программа

Наша программа должна выводить в командной строке приветствие.

Смысл приветствия «Hello, World!» переводится буквально как «Привет, мир!» Но лучше узнать правильный перевод.

Задание. Просмотрите в Википедии статью **Hello, world!** Выясните, как правильно переводится это сообщение.

Творческое задание. Просмотрите в Википедии статью **stdio. h**.

Чтобы скомпилировать нашу программу и получить исполняемый файл, вызываем компилятор и сообщаем ему название файла с исходным текстом программы, см. рис.

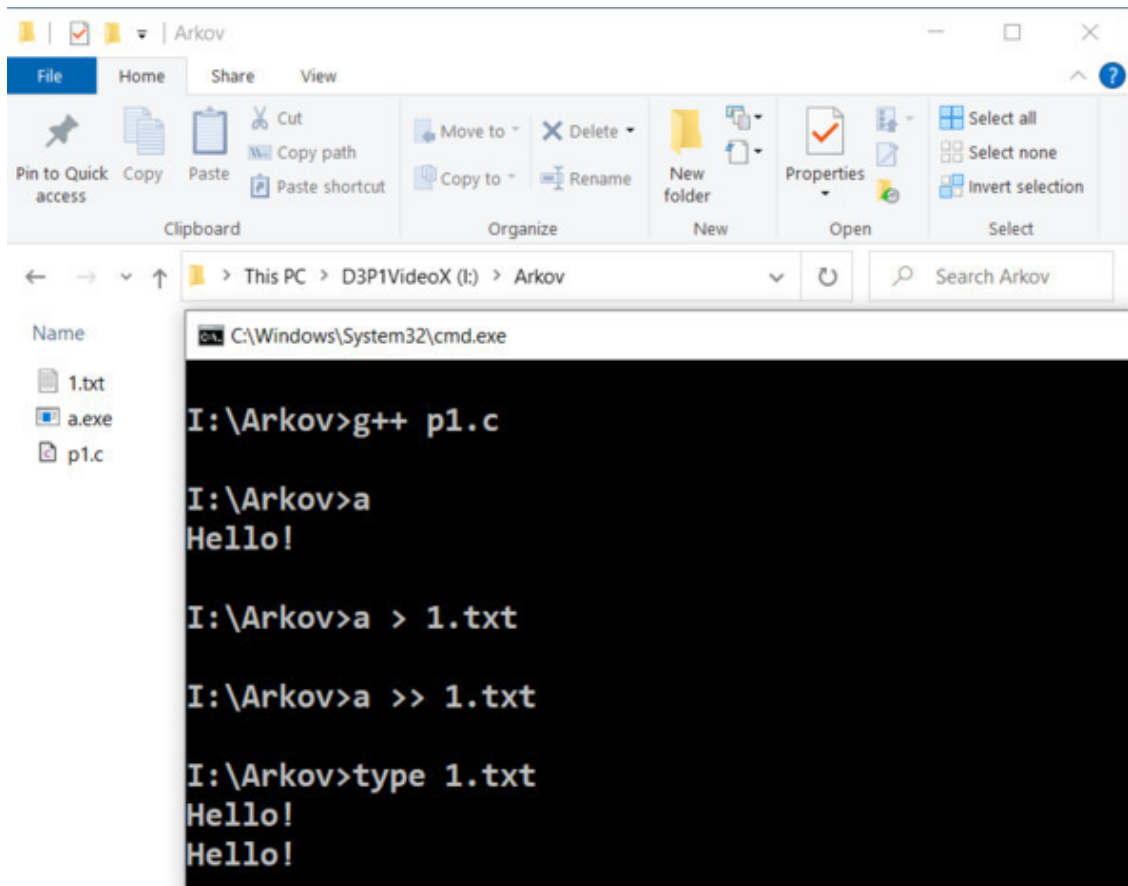


Рис. Компиляция первой программы

Компилятор действует по умолчанию и создаёт исполняемый файл под названием **a.exe**. Мы запускаем этот файл на выполнение – расширение **EXE** можно не указывать. Убеждаемся, что он выводит своё сообщение на экран.

Затем мы перенаправляем вывод в файл. В следующей команде добавляем новую строку в существующий файл.

Наконец, мы выводим содержимое файла на экран командой **TYPE**.

Задание. Скомпилируйте свою программу и повторите описанный опыт.

Чтобы задать другое название для исполняемого файла, используем опцию **-o**, см. рис.

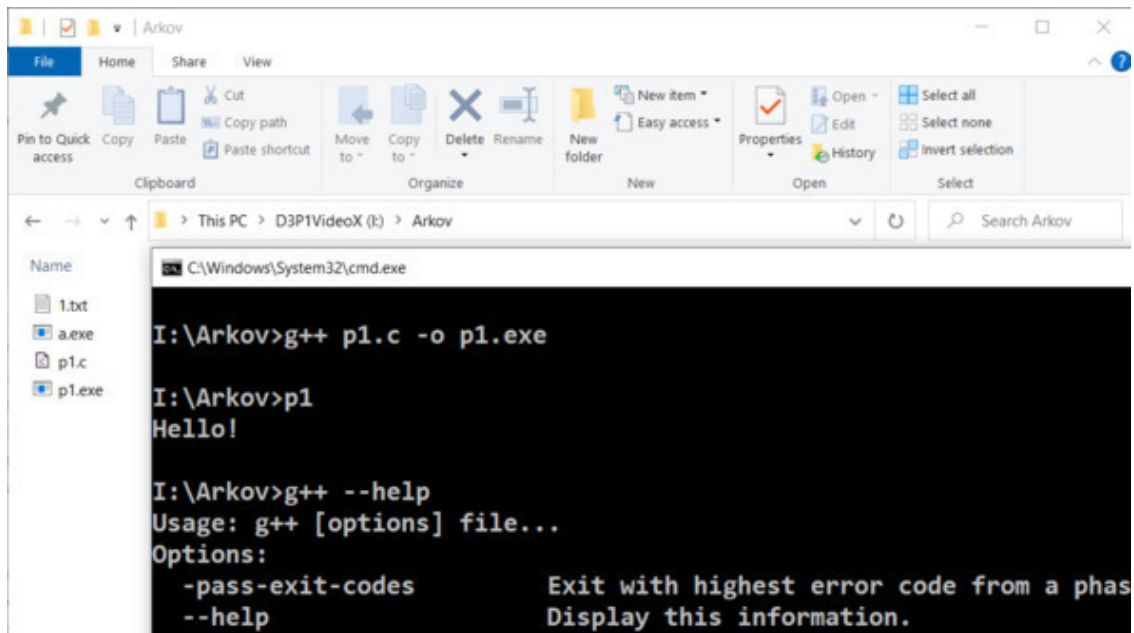


Рис. Дополнительные параметры компилятора

Для получения справки о параметрах компилятора вызываем его с опцией – **help**.

Задание. Скомпилируйте свою программу с указанием имени выходного файла, а затем запустите его.

Задание. Просмотрите список параметров компилятора и выясните, какое английское слово скрывается в опции **-o**.

Творческое задание. Выясните, что означает название команды **g++**.

Программа с вводом с клавиатуры

Теперь составим программу, которая читает данные с клавиатуры – устройства стандартного ввода.

Используем команду **scanf** для чтения числа, которое вводится с клавиатуры.

Затем выводим квадрат полученного числа командой **printf**.

Наша программа будет называться **p2**, см. рис.

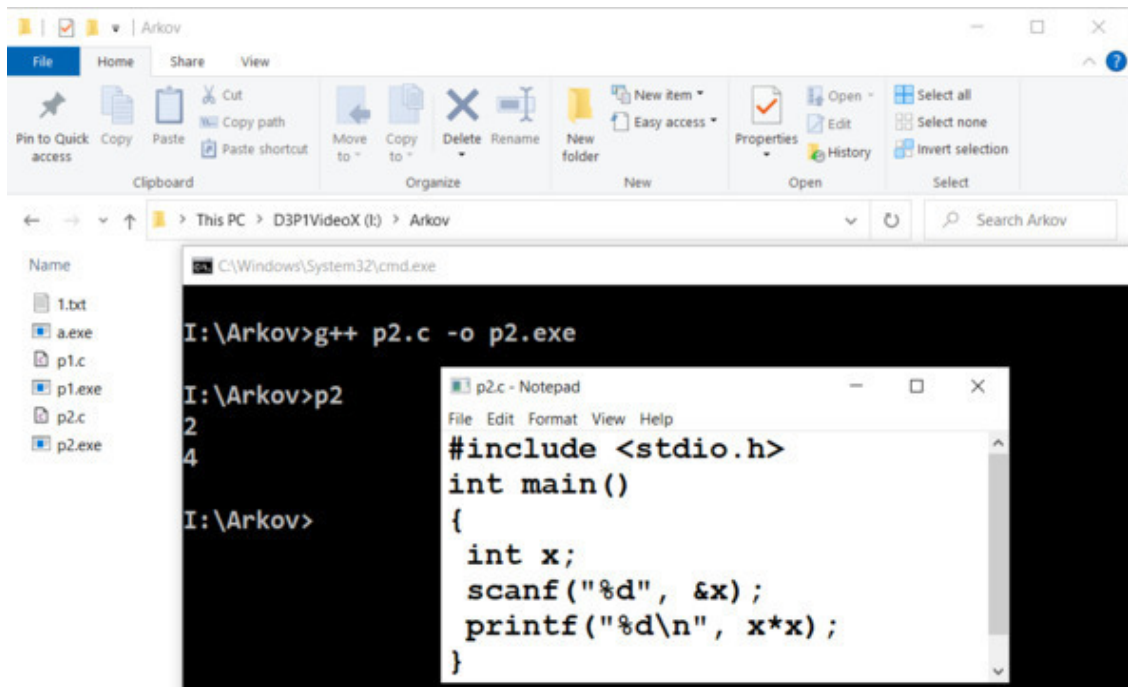


Рис. Программа ввода с клавиатуры

Далее используем перенаправление стандартного ввода. Пусть программа прочитает входное число из файла.

Создаём файл и записываем в него число.

Вызываем программу с указанием входного файла.

Получаем вывод результата на экран, см. рис.

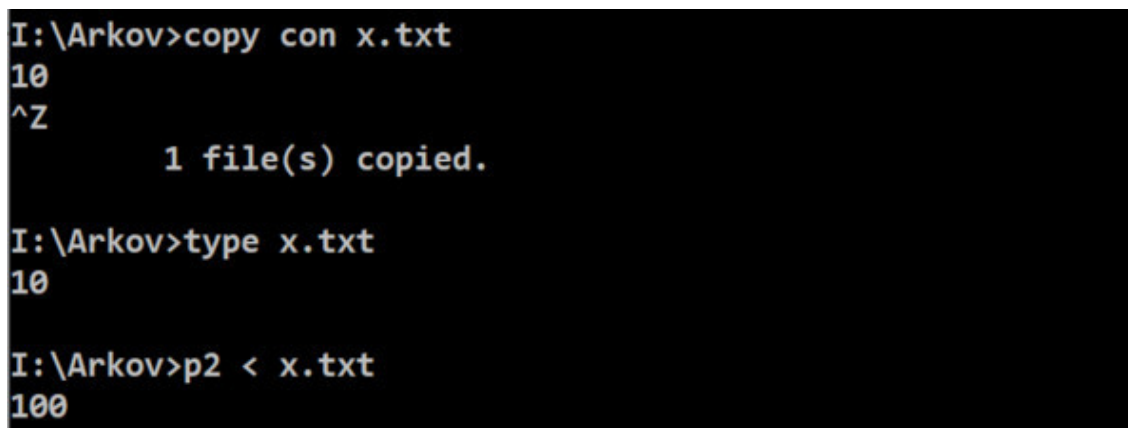


Рис. Перенаправление ввода

Задание. Проведите описанный опыт.

Мы можем сохранить результаты в файл с помощью перенаправления стандартного вывода.

Можем одновременно использовать входные данные из файла и отправить результаты в файл, см. рис.

```
I:\Arkov>p2 > y.txt
5

I:\Arkov>type y.txt
25

I:\Arkov>p2 < x.txt > z.txt

I:\Arkov>type z.txt
100
```

Рис. Перенаправление ввода и вывода

Задание. Проведите описанный опыт.

В предыдущих опытах мы убедились, что перенаправление ввода-вывода работает. Теперь можно организовать конвейер в командной строке. Наша программа **p2** читает поток стандартного ввода и пишет в поток стандартного вывода. Объединим несколько вызовов одной и той же программы в конвейер, см. рис.

```
I:\Arkov>p2 | p2
10
10000

I:\Arkov>p2 | p2 | p2 | p2
2
65536
```

Рис. Конвейер

Задание. Проведите описанный опыт.

Программа с чтением параметров командной строки

Теперь мы составим программу, которая использует параметры (аргументы) командной строки. Это не то же самое, что стандартный ввод-вывод. В данном случае параметры вводят через пробел после имени программы.

Для работы с аргументами командной строки в заголовке основного модуля программы получаем количество аргументов **argc** – *arguments count*. Вторым параметром **argv** – это массив текстовых строк *arguments vector* – значения аргументов командной строки.

Для начала выводим на экран значение нулевого аргумента. Это имя файла, которое использовано для запуска программы, см. рис.

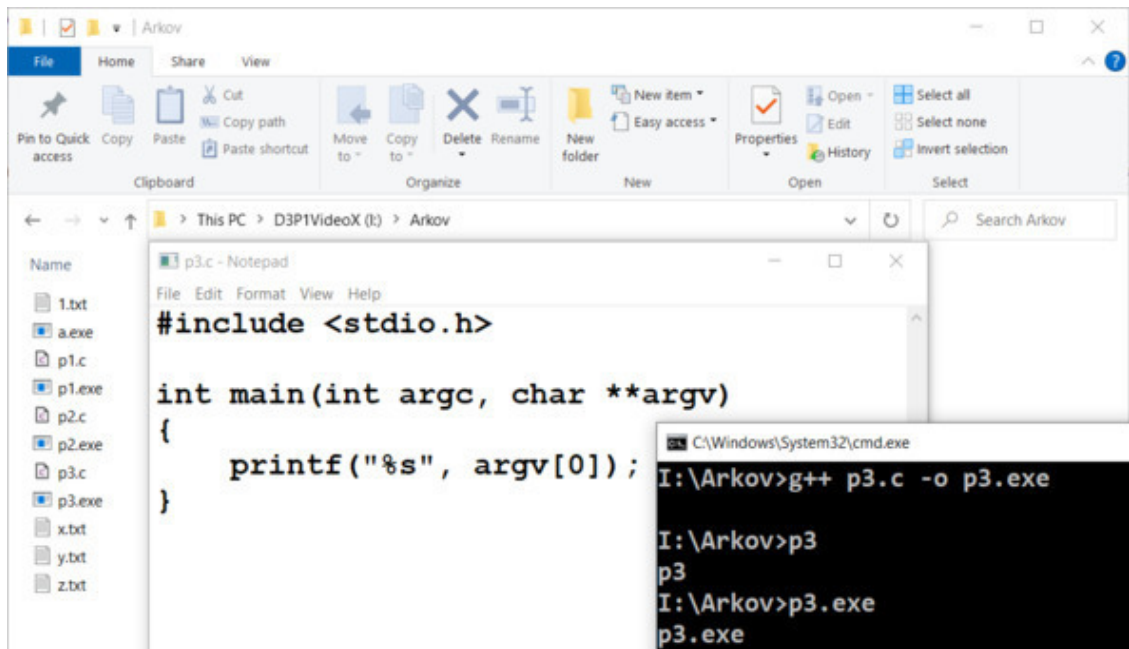


Рис. Нулевой параметр командной строки

Задание. Проведите описанный опыт.

Творческое задание. Просмотрите в Википедии английскую статью **Entry point** и выясните, что ещё можно сообщить программе при объявлении основного модуля **main**.

Продолжаем эксперименты. Наша четвёртая программа читает целое число как аргумент команды и выводит на экран квадрат этого числа, см. рис.

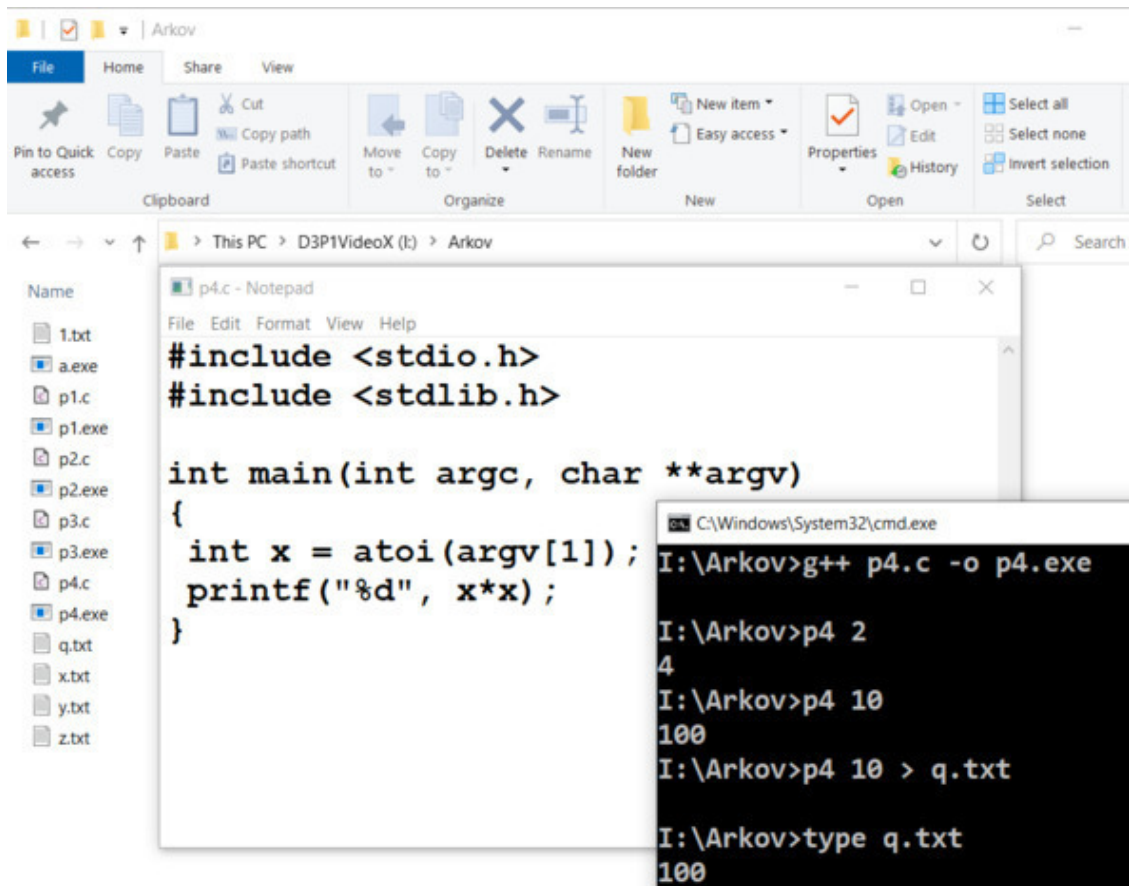


Рис. Чтение числа как аргумента командной строки

Здесь придётся использовать функцию **atoi** для преобразования строки в число.

Название функции можно расшифровать таким образом:

a – строка (буква **a** в противоположность цифрам)

to – преобразовать в

i – целое число (integer).

Для использования функции **atoi** нужно подключить стандартную библиотеку **stdlib**.

Задание. Проведите описанный опыт.

Творческое задание. Подберите такие значения аргумента, которые приводят к ошибкам в расчётах.

Файлы и каталоги

Файлы для операционной системы – это программы (исполняемые файлы) и данные (с которыми работают команды и программы).

Файлы и каталоги организованы в файловое дерево.

Команды операционной системы позволяют выводить на экран информацию о файлах и каталогах, а также создавать и уничтожать, копировать, переносить и переименовывать их.

DIR – Содержимое каталога

Итак, наш текстовый файл записан, и мы даже можем заглянуть внутрь него. Посмотрим, какая информация по нему доступна. Вызываем команду **dir**, см. рис.

```

Командная строка
C:\Users\Valentin>dir
Том в устройстве C не имеет метки.
Серийный номер тома: 1433-C702

Содержимое папки C:\Users\Valentin

10.09.2021  10:53      <DIR>          .
10.09.2021  10:53      <DIR>          ..
19.09.2015  14:57      <DIR>          .FBReader
11.06.2021  18:12      <DIR>          .gimp-2.8
16.04.2015  16:12      <DIR>          .thumbnails
09.11.2019  13:29      <DIR>          .VirtualBox
01.02.2019  13:41             46 1.BAT
01.02.2019  13:41             46 1.cmd
28.06.2021  15:39      <DIR>          3D Objects
28.06.2021  16:10      <DIR>          ansel
01.02.2019  12:55             122 arg.7z
01.02.2019  13:56              7 c.bat
10.09.2021  10:53             6 221 commands.txt
28.06.2021  15:39      <DIR>          Contacts
  
```

Рис. Содержимое каталога

Команда **dir** выводит на экран список файлов и каталогов.

По умолчанию работает с текущим каталогом.

Название команды **dir** – это сокращение от слова **directory** – «каталог».

Можно встретить самые названия у этого загадочного объекта, внутри которого расположены файлы:

- directory;
- директория;
- folder;
- папка;
- каталог...

Все эти слова означают одно и то же. Даже в рамках одной ОС могут использоваться несколько разных названий.

Надо понимать, что операционная система – это большой комплекс программ. Это десятки миллионов строк кода. Это работа нескольких тысяч программистов. Это десятки лет

почти круглосуточной работы. И несколько сотен переводчиков тоже приложили к этому проекту руку.

Так что неудивительно, что здесь появляются разные названия. В подобной ситуации один конструктор самолётов сказал: «Я удивляюсь, как они после этого вообще летают».

Творческое задание. Просмотрите в Википедии статью **Количество строк кода** и выясните ориентировочное число строк кода в ОС Windows.

Кроме названия файла или каталога в этом списке можно найти размер файла в байтах, а также дату и время его создания.

Каталоги в этом списке обозначены символами **<DIR>** слева от названия. Рядом с названием файла не указано ничего, зато есть его размер.

Задание. Уточните произношение и перевод слова **directory** в **Переводчиках Яндекс** и **Google**.

Задание. Вызовите команду **dir** и найдите свой текстовый файл в полученном списке. Определите размер своего файла.

Команда **dir** может вывести список файлов в самых разных форматах и в разном порядке.

Чтобы узнать подробности, можно почитать справку по данной команде, см. рис.

```

Командная строка
C:\Users\Valentin>dir /?
Вывод списка файлов и подкаталогов в указанном каталоге.

DIR [drive:][path][filename] [/A[[:]attributes]] [/B] [/C] [/D] [/L] [/N]
  [/O[[:]sortorder]] [/P] [/Q] [/R] [/S] [/T[[:]timefield]] [/W] [/X] [/4]

  [drive:][path][filename]
      Диск, каталог или имена файлов для включения в список.
  
```

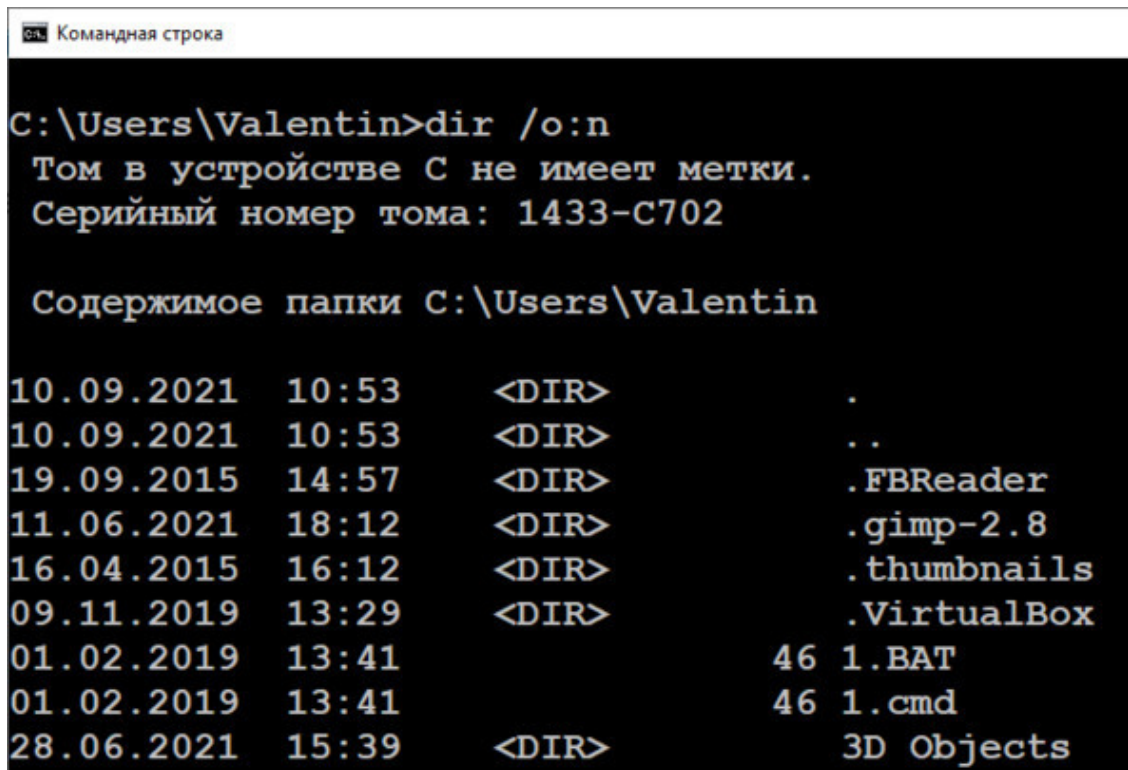
Рис. Параметры команды **dir**

Как видим, здесь очень много квадратных скобок, но это не страшно. Напомним, что квадратные скобки показывают необязательные аргументы команды. То есть можно не указывать всё, что в квадратных скобках. Можно использовать только отдельные параметры или их комбинации.

И не надо в командах писать сами квадратные скобки. Это для читателя, а не для компьютера. Если на защите лабораторной работы студент читает справку по команде и печатает квадратные скобки, значит он первый раз в жизни видит командную строку.

Однако, вернёмся нашей команде **dir**.

Мы можем с помощью дополнительных параметров отсортировать список файлов по алфавиту, см. рис.



```

C:\Users\Valentin>dir /o:n
Том в устройстве C не имеет метки.
Серийный номер тома: 1433-C702

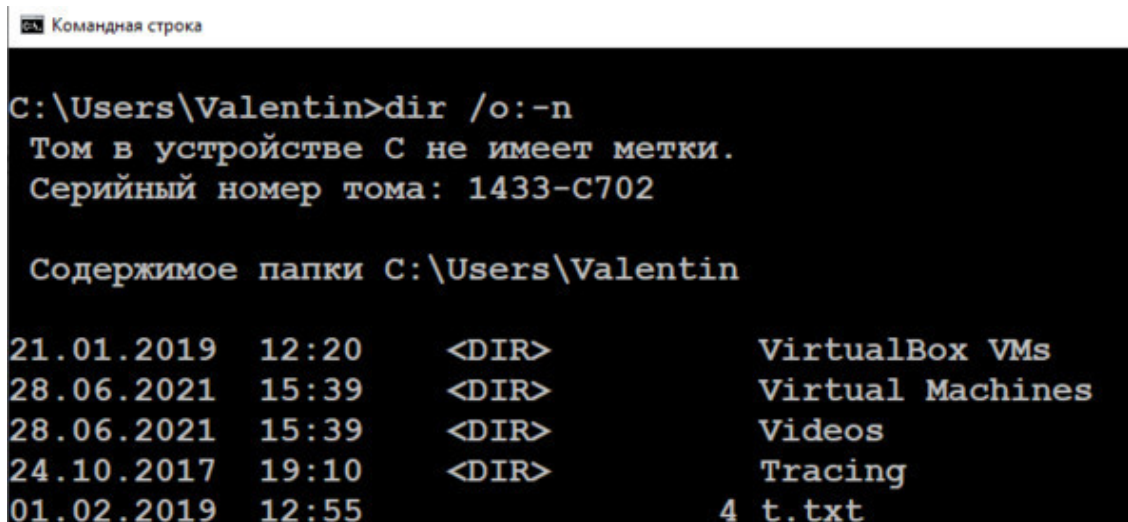
Содержимое папки C:\Users\Valentin

10.09.2021  10:53      <DIR>          .
10.09.2021  10:53      <DIR>          ..
19.09.2015  14:57      <DIR>          .FBReader
11.06.2021  18:12      <DIR>          .gimp-2.8
16.04.2015  16:12      <DIR>          .thumbnails
09.11.2019  13:29      <DIR>          .VirtualBox
01.02.2019  13:41             46 1.BAT
01.02.2019  13:41             46 1.cmd
28.06.2021  15:39      <DIR>          3D Objects

```

Рис. Список в алфавитном порядке

Можно провести сортировку в обратном порядке, см. рис.



```

C:\Users\Valentin>dir /o:-n
Том в устройстве C не имеет метки.
Серийный номер тома: 1433-C702

Содержимое папки C:\Users\Valentin

21.01.2019  12:20      <DIR>          VirtualBox VMs
28.06.2021  15:39      <DIR>          Virtual Machines
28.06.2021  15:39      <DIR>          Videos
24.10.2017  19:10      <DIR>          Tracing
01.02.2019  12:55             4 t.txt

```

Рис. Сортировка по алфавиту в обратном порядке

Параметры (Parameters) или аргументы (Arguments) команды могут также называть опциями (options), ключами (switches) и другими словами. Опять же всё зависит от версии ОС, команды разработчиков и опыта переводчиков.

Параметры команды **dir** – это отдельные буквы, но это по-прежнему начало какого-нибудь английского слова.

Например, сортировка – параметр **o** – начало слова *order* – «упорядочить, отсортировать в указанном порядке».

Сортировка по алфавиту – **n** – от слова *name* – «название файла или каталога».

Задание. Вызовите справку для команды **dir** и ознакомьтесь с её возможностями.

Задание. Выведите на экран только названия файлов – без дополнительной информации.

Задание. Выясните, что делает команда **dir /s** и запустите её на выполнение. Чтобы прервать вывод на экран, нажмите комбинацию клавиш **[Ctrl + C]**.

Задание. Выясните, что делает команда **dir /x** и запустите её на выполнение.

Творческое задание. Просмотрите на Википедии статью **8.3**:

<https://ru.wikipedia.org/wiki/8.3>

и выясните, как и для чего используется формат **8dot3**. Здесь слово **dot** означает «точка».

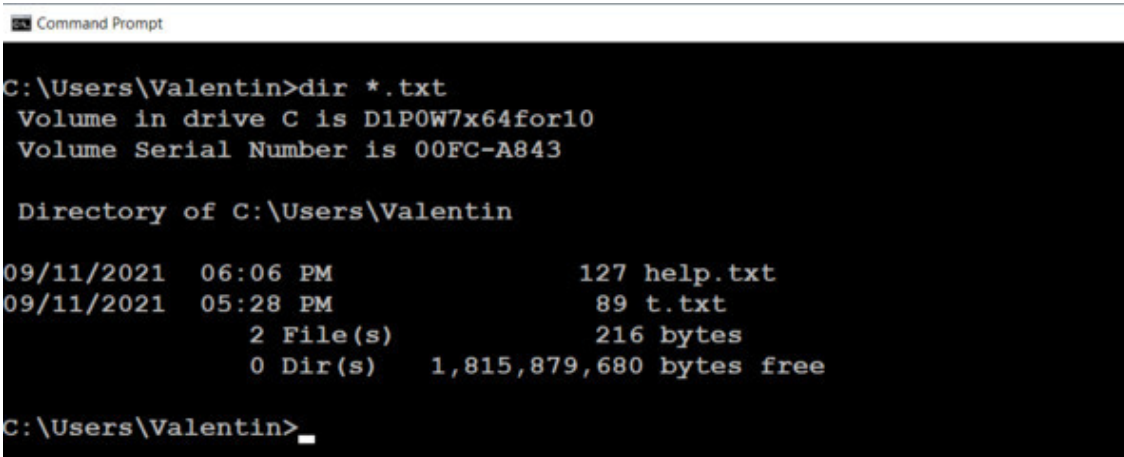
Можно вывести на экран не весь список файлов, а только то, что нас интересует. Для этого используют так называемые шаблоны. Обычно про них рассказывают в школе на уроках информатики, но мы напомним основную идею.

Чтобы получить список всех файлов с расширением **txt**, вводим команду **dir *.txt**, см. рис.

Здесь символ звёздочки означает любое количество символов в названии файла и даже их отсутствие. Будут выбраны файлы, название которых заканчивается на точку и буквы **txt**.

Ещё можно использовать знак вопроса. Вместо него можно подставить только один символ.

Задание. Просмотрите в Википедии англоязычную статью **Wildcard character** и соответствующую русскоязычную статью.



```
Command Prompt

C:\Users\Valentin>dir *.txt
Volume in drive C is D1P0W7x64for10
Volume Serial Number is 00FC-A843

Directory of C:\Users\Valentin

09/11/2021  06:06 PM                127 help.txt
09/11/2021  05:28 PM                 89 t.txt
                2 File(s)                216 bytes
                0 Dir(s)  1,815,879,680 bytes free

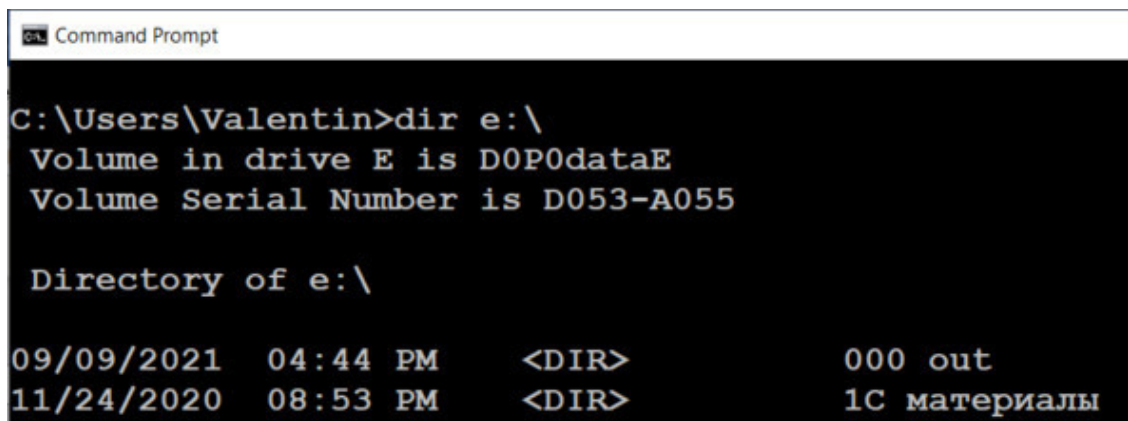
C:\Users\Valentin>
```

Рис. Использование шаблона

Задание. Выведите список файлов с расширением **txt**.

Задание. Выведите список файлов, название которых начинается на букву **a**.

Мы можем указать диск или каталог, который нас интересует, см, рис.



```
Command Prompt

C:\Users\Valentin>dir e:\
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

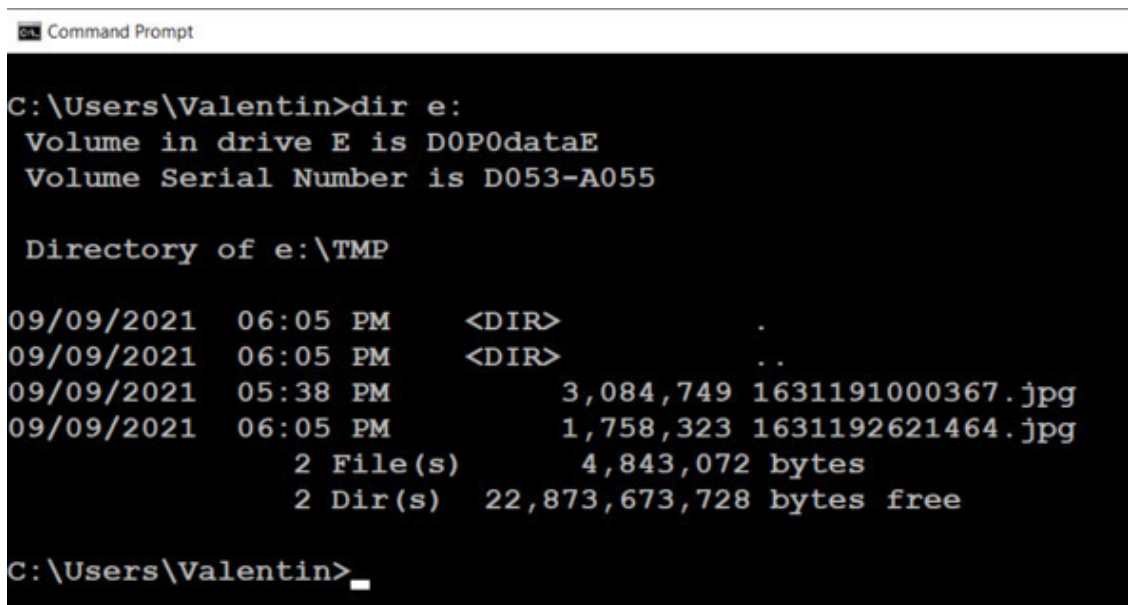
Directory of e:\

09/09/2021  04:44 PM    <DIR>          000 out
11/24/2020  08:53 PM    <DIR>          1C материалы
```

Рис. Содержимое корневого каталога диска

Название диска в системе Windows – это латинская буква и двоеточие. Обратная наклонная черта указывает корневой каталог выбранного диска.

Если не указывать каталог, то команда **dir** будет работать с текущим каталогом указанного диска. Здесь результат может быть непредсказуемым, см. рис.



```
Command Prompt

C:\Users\Valentin>dir e:
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of e:\TMP

09/09/2021  06:05 PM    <DIR>          .
09/09/2021  06:05 PM    <DIR>          ..
09/09/2021  05:38 PM             3,084,749 1631191000367.jpg
09/09/2021  06:05 PM             1,758,323 1631192621464.jpg
                2 File(s)          4,843,072 bytes
                2 Dir(s)  22,873,673,728 bytes free

C:\Users\Valentin>
```

Рис. Содержимое текущего каталога диска

Задание. Вызовите команду **DIR** для любого диска, кроме текущего. Обратите внимание на текущий каталог.

STDERR – Поток сообщений об ошибках

В предыдущих упражнениях мы познакомились со стандартными потоками ввода-вывода **STDIN** и **STDOUT**. В операционных системах есть ещё один инструмент – поток сообщений об ошибках **STDERR**.

При перенаправлении ввода-вывода сообщения об ошибках по-прежнему выводятся на экран.

Чтобы перехватывать эти сообщения и отправить их в файл, используют особые комбинации символов, см. рис.

```
dir x
dir x > 1
dir x > 1 2>2
dir x > 1.txt 2>&1.txt
```

Рис. Перенаправление сообщений об ошибках

В этих примерах мы указываем название файла, которого нет в текущем каталоге. В этом случае команда **DIR** вначале выводит общую информацию о диске в поток стандартного вывода, а затем сообщение об ошибке – в поток сообщений об ошибках.

В первой строке примера мы получаем все сообщения на экране.

Во второй строке мы перенаправляем стандартный вывод в файл, а сообщение об ошибке выводится на экран.

В третьей строке мы перенаправляем сообщение об ошибке в отдельный файл.

В четвёртой строке мы отправляем все сообщения в один общий файл.

Задание. Выполните приведённые выше команды, обращая внимания на пробелы между символами. После каждой команды проверьте содержимое созданных текстовых файлов.

Творческое задание. Просмотрите в Википедии статью **Стандартные потоки**.

CD – Текущий каталог

В операционной системе Windows используются понятия текущего диска и текущего каталога.

Если не указывать диск и каталог, то команды ОС работают с текущим каталогом на текущем диске. Это так называемые «действия по умолчанию».

Кроме того, для каждого диска назначается свой текущий каталог.

В каждом каталоге могут находиться файлы и другие (вложенные) каталоги – подкаталоги. Вложенные каталоги образуют дерево. Его ещё называют файловым деревом, см. рис.

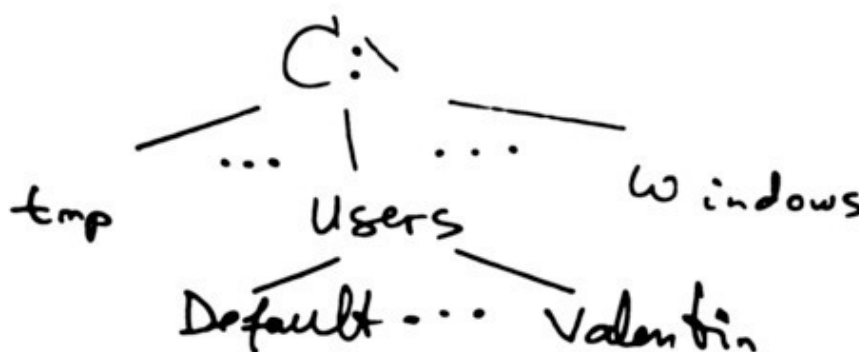


Рис. Дерево каталогов

Плюс к этому, командное окно начинает работу с домашнего каталога пользователя, см. рис.

```
cmd
Командная строка

C:\Users\Valentin>cd
C:\Users\Valentin

C:\Users\Valentin>prompt $G

>cd
C:\Users\Valentin

>
```

A screenshot of a Windows Command Prompt window titled 'Командная строка'. The window shows the following sequence of commands and output:
1. Initial prompt: `C:\Users\Valentin>`
2. Command: `cd`
3. Output: `C:\Users\Valentin`
4. Command: `prompt $G`
5. Prompt changes to `>`
6. Command: `cd`
7. Output: `C:\Users\Valentin`
8. Prompt changes to `>`

Рис. Текущий каталог

Команда **CD** без параметров просто выводит на экран текущий диск и каталог.

По умолчанию эти сведения уже выводятся в приглашении к вводу команд. Вроде бы получается дублирование информации...

Представим, что текущий каталог – это длинная строка на половину экрана. С таким длинным приглашением будет неудобно работать. В этом случае придётся организовать при-

глашение покороче. Но тогда пропадают из виду сведения о текущем диске и каталоге. Придётся время от времени их вызывать на экран. Тут-то нам и поможет команда **CD**.

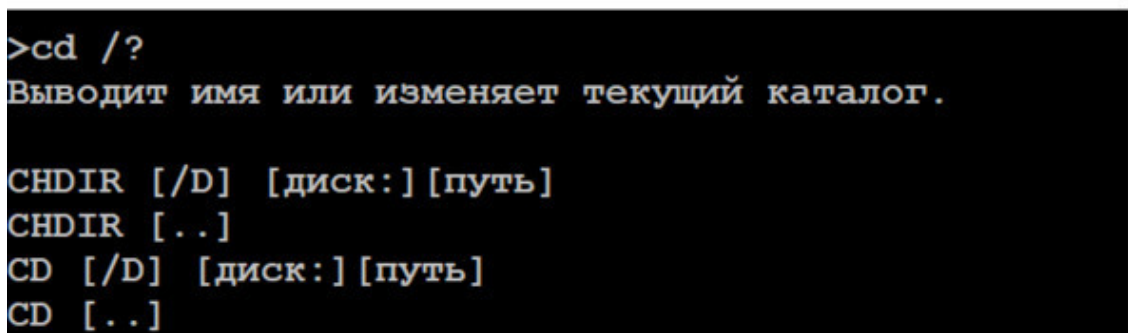
В следующем упражнении нам понадобятся две новые команды:

CD – Change Directory – Сменить каталог

CLS – Clear Screen – Очистить экран

Задание. Запустите командное окно. Обратите внимание на информацию в приглашении к вводу команд. Настройте короткое приглашение командой **PROMPT \$G**. Очистите экран командой **CLS**. Вызовите команду **CD** без параметров.

Судя по названию, команда **CD** предназначена для смены каталога. Посмотрим описание команды, см. рис.



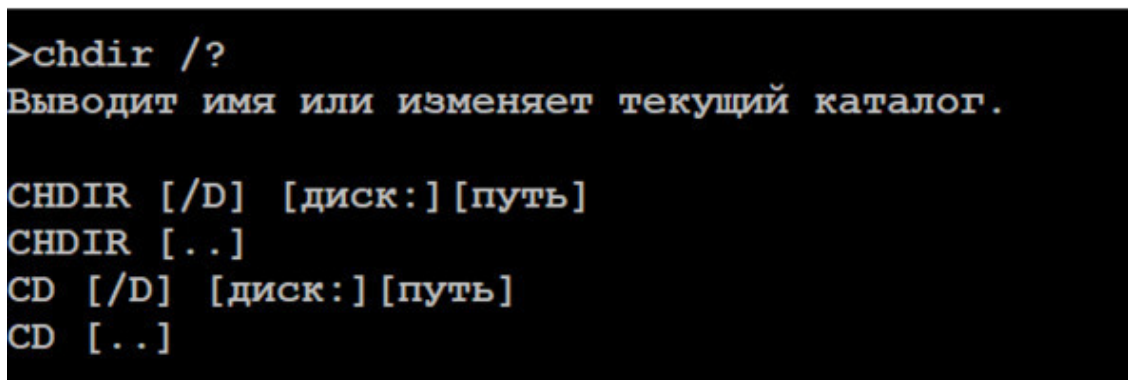
```
>cd /?
Выводит имя или изменяет текущий каталог.

CHDIR [/D] [диск:] [путь]
CHDIR [...]
CD [/D] [диск:] [путь]
CD [...]
```

Рис. Описание команды **CD**

Как видим, эта команда может просто вывести текущий каталог, а может и изменить его.

Кроме того, мы обнаруживаем, что эту же команду можно вызвать и по более длинному названию **CHDIR**. Вызовем справку по **CHDIR** и обнаружим, что это одна и та же команда, см. рис.



```
>chdir /?
Выводит имя или изменяет текущий каталог.

CHDIR [/D] [диск:] [путь]
CHDIR [...]
CD [/D] [диск:] [путь]
CD [...]
```

Рис. Описание команды **CHDIR**

Итак, поведение команды **CD** не всегда интуитивно понятно.

Команда **CD** без параметров выводит текущий каталог.

Команда **CD** с указанием диска выводит текущий каталог указанного диска, см. рис.

```
C:\Users\Valentin>cd
C:\Users\Valentin

C:\Users\Valentin>cd i:
I:\

C:\Users\Valentin>cd..

C:\Users>cd valentin

C:\Users\Valentin>cd \

C:\>_
```

Рис. Текущий каталог

Команда **CD..** позволяет перейти в каталог более высокого уровня по иерархии. В нашем примере это переход из каталога **C:\Users\Valentin** в каталог **C:\Users**.

Команда **CD ** – переход в корневой каталог текущего диска.

Для перехода на другой диск (смена текущего диска) потребуется опция **/D**. Есть и более короткий вариант – просто указать букву диска и двоеточие, см. рис.

```
C:\>CD /D L:\TMP

L:\TMP>e:

E:\>i:

I:\>L:

L:\TMP>cd /d e:\arkov

e:\Arkov>_
```

Рис. Смена текущего диска

Задание. Потренируйтесь в использовании команды **CD**.

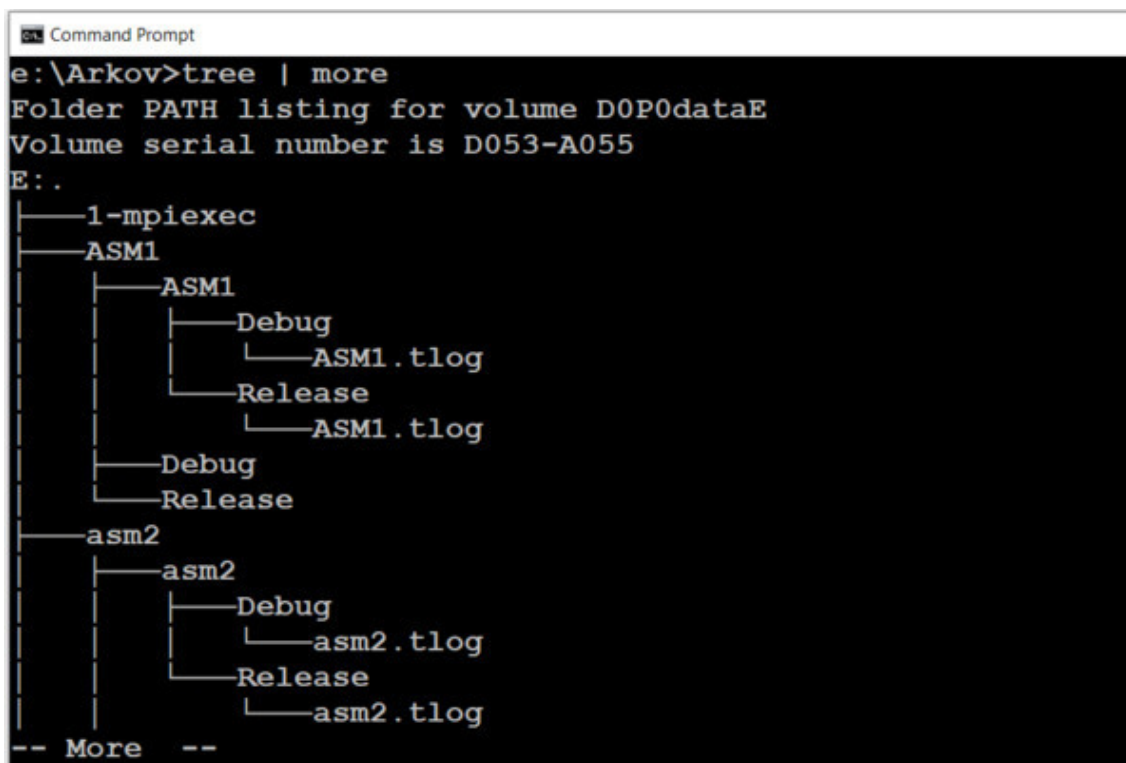
TREE – Файловое дерево

Дерево каталогов можно попытаться изобразить в командной строке. Однако, здесь можно выводить только символы: буквы, цифры и «псевдографику» (символы, содержащие линии).

Команда **TREE** позволяет вывести на экран дерево каталогов диска – начиная с корневого или указанного каталога.

Если дерево занимает много страниц, лучше сразу же использовать команду **MORE** для постраничного вывода.

Вывод на экран можно остановить с помощью комбинации клавиш **[Ctrl + C]** или **[Ctrl + Break]**. В этом случае на экране появится строчка **^C**.



```
e:\Arkov>tree | more
Folder PATH listing for volume D0P0dataE
Volume serial number is D053-A055
E:..
|-- 1-mpiexec
|-- ASM1
|   |-- ASM1
|   |   |-- Debug
|   |   |   |-- ASM1.tlog
|   |   |-- Release
|   |       |-- ASM1.tlog
|   |-- Debug
|   |-- Release
|-- asm2
|   |-- asm2
|   |   |-- Debug
|   |   |   |-- asm2.tlog
|   |   |-- Release
|   |       |-- asm2.tlog
-- More --
```

Рис. Дерево каталогов

Здесь мы увидим только структуру каталогов, без списка файлов.

Задание. Повторите показанные опыты.

Задание. Просмотрите справку для команды **TREE** и выясните, как вывести список файлов. Проверьте, как работает эта опция.

Творческое задание. Просмотрите в Википедии английскую статью **tree (command)**.

COPY – Копирование файлов

Команда **COPY** позволяет скопировать один или несколько файлов в указанное место или под указанным именем. Здесь можно указать место предназначения для копируемых файлов или путь и название нового файла.

Задание. Изучите справку для команды **COPY** и выясните, как объединить несколько файлов в один.

Задание. Создайте файл **1.txt** и скопируйте его в файл **2.txt**. Повторите операцию копирования и обратите внимание на предупреждение ОС.

Задание. Скопируйте два файла **1.txt** и **2.txt** так, чтобы объединить их в один файл **3.txt**.

Задание. Скопируйте файл **1.txt** в корневой каталог диска **D:**.

Задание. Скопируйте все файлы с расширением **TXT** в корневой каталог диска **D:**.

MOVE – Перенести файлы

Команда MOVE позволяет перенести (переместить) файлы. Здесь можно указать место расположения исходных файлов, а также их место предназначения и название.

Задание. Изучите справку для команды **MOVE** и выясните, что ещё можно сделать с помощью данной команды.

Задание. Переместите файл **1.txt** в корневой каталог диска **D:** под названием **4.txt**.

DEL – Удаление файла

Команда **DEL** позволяет удалить один или несколько файлов.

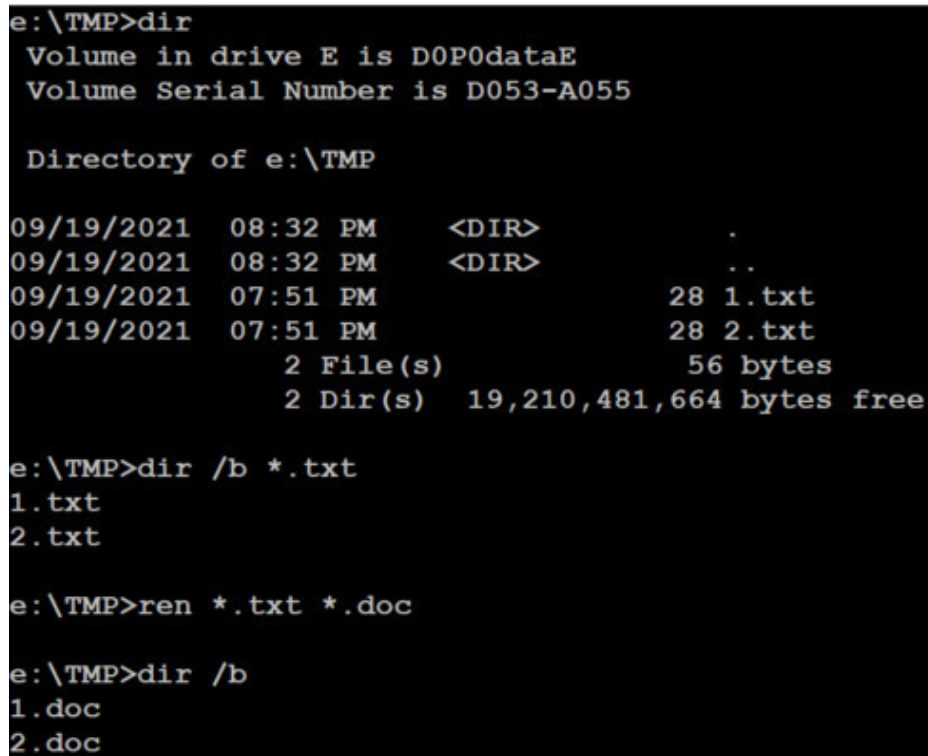
Задание. Изучите справку для команды **DEL** и выясните, как удалять файлы в подкаталогах и файлы «только для чтения».

RENAME – Переименование файла

Команда **RENAME** позволяет переименовать один или несколько файлов. При этом файлы остаются в своём каталоге.

Обратим внимание, что данная команда не может переименовать каталог. Для этого служит одна из предыдущих команд.

Для работы с несколькими файлами можно использовать символ «звёздочку», см. рис.



```
e:\TMP>dir
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of e:\TMP

09/19/2021  08:32 PM    <DIR>          .
09/19/2021  08:32 PM    <DIR>          ..
09/19/2021  07:51 PM                28 1.txt
09/19/2021  07:51 PM                28 2.txt
               2 File(s)                56 bytes
               2 Dir(s)  19,210,481,664 bytes free

e:\TMP>dir /b *.txt
1.txt
2.txt

e:\TMP>ren *.txt *.doc

e:\TMP>dir /b
1.doc
2.doc
```

Рис. Переименование файлов

Задание. Изучите справку для команды **RENAME**. Обратите внимание на короткий псевдоним для команды.

MKDIR – Создание каталога

Команда **MKDIR** позволяет создать каталог. Если задать путь к каталогу, команда может создать недостающие подкаталоги.

Задание. Изучите справку для команды **RENAME**. Обратите внимание на короткий псевдоним для команды.

Задание. Предположите, что произойдёт при выполнении следующих команд:

MKDIR 1

MKDIR \1

MKDIR 1\1

MKDIR \1\1

Введите команды по очереди и проверьте свои предположения.

RMDIR – Удаление каталога

Команда **RMDIR** позволяет удалить каталог.

Задание. Изучите справку для команды **RMDIR**. Обратите внимание на короткий псевдоним для команды.

Творческое задание. Выясните, как работает команда **RD** с ключами **/S** и **/Q**.

ATTRIB – Атрибуты и разрешения

Для каждого файла можно задать атрибуты. Это свойства файла, которые чаще всего определяют права доступа – разрешения на операции с файлами – permissions.

Основные атрибуты файлов:

R – Read-only – Файл только для чтения

A – Archive – Файл готов для архивирования

S – System – Системный файл

H – Hidden – Скрытый файл

Самый загадочный из атрибутов – «Архивный». Стандартное объяснение про «готовность к архивированию» мало что объясняет.

Задание. Выясните, для чего на самом деле может использоваться атрибут «Архивный».

Задание. Изучите справку для команды **ATTRIB**. Выясните, как работать с атрибутами файлов в подкаталогах и с атрибутами каталогов.

Команда **ATTRIB** без параметров выводит список файлов с указанием их атрибутов, см. рис.

```
e:\TMP>copy con 1.txt
1
^Z
1 file(s) copied.

e:\TMP>copy 1.txt 2.txt
1 file(s) copied.

e:\TMP>copy 1.txt 3.txt
1 file(s) copied.

e:\TMP>attrib
A          E:\TMP\1.txt
A          E:\TMP\2.txt
A          E:\TMP\3.txt
```

Рис. Просмотр атрибутов

Чтобы установить атрибут, используем символ «плюс», чтобы снять атрибут – символ «минус».

Некоторые атрибуты лишают пользователя права на удаление файла, см. рис.

```

e:\TMP>attrib +s 1.txt

e:\TMP>attrib +h 2.txt

e:\TMP>attrib +r 3.txt

e:\TMP>attrib
A S          E:\TMP\1.txt
A H          E:\TMP\2.txt
A R          E:\TMP\3.txt

e:\TMP>del 1.txt
Could Not Find e:\TMP\1.txt

e:\TMP>del 2.txt
Could Not Find e:\TMP\2.txt

e:\TMP>del 3.txt
e:\TMP\3.txt
Access is denied.

e:\TMP>del /f 3.txt

e:\TMP>

```

Рис. Изменение атрибутов

Задание. Повторите описанный выше опыт.

Задание. Выясните, что делает команда **DEL /F**.

Для работы с несколькими файлами можно использовать шаблоны типа *.*.

```

e:\TMP>md 1

e:\TMP>attrib
A S          E:\TMP\1.txt
A H          E:\TMP\2.txt

e:\TMP>attrib -s -h *.*

e:\TMP>attrib
A          E:\TMP\1.txt
A          E:\TMP\2.txt

e:\TMP>attrib /d
A          E:\TMP\1
A          E:\TMP\1.txt
A          E:\TMP\2.txt

e:\TMP>attrib +s /d *.*

e:\TMP>attrib /d
S          E:\TMP\1
A S        E:\TMP\1.txt
A S        E:\TMP\2.txt

e:\TMP>

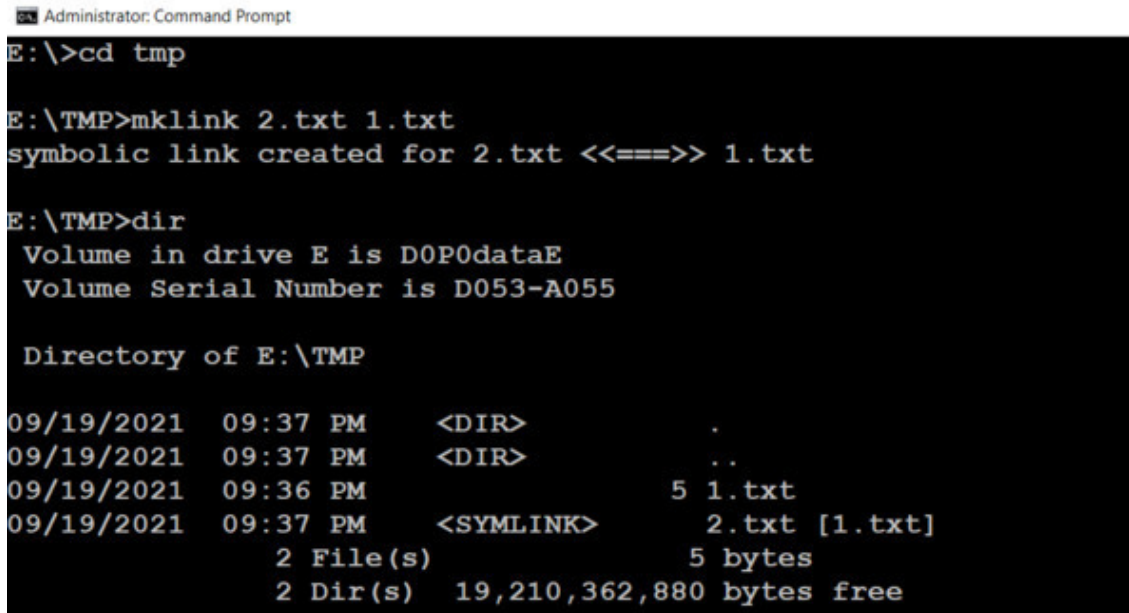
```

Рис. Изменение атрибутов для нескольких объектов

Задание. Снимите установленные атрибуты **S**, **H**, **R** для файлов и каталогов. Удалите файлы.

MKLINK – Ссылка на файл

Команда **MKLINK** позволяет создать ссылку на файл. Для этого может потребоваться запустить командное окно с правами администратора, см. рис.



```
Administrator: Command Prompt
E:\>cd tmp

E:\TMP>mklink 2.txt 1.txt
symbolic link created for 2.txt <==> 1.txt

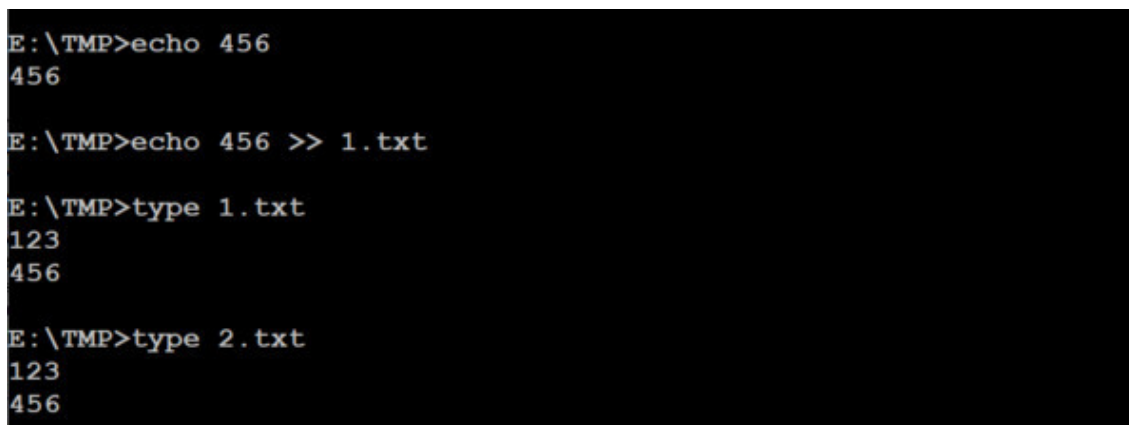
E:\TMP>dir
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of E:\TMP

09/19/2021  09:37 PM    <DIR>          .
09/19/2021  09:37 PM    <DIR>          ..
09/19/2021  09:36 PM                5 1.txt
09/19/2021  09:37 PM    <SYMLINK>      2.txt [1.txt]
                2 File(s)                5 bytes
                2 Dir(s)  19,210,362,880 bytes free
```

Рис. Создание ссылки на файл

Добавим новую строку в исходный файл. Используем команду **ЕCHO** для вывода строки текста на экран или в файл – если перенаправить поток стандартного вывода. Файл-ссылка тоже изменится, см. рис.



```
E:\TMP>echo 456
456

E:\TMP>echo 456 >> 1.txt

E:\TMP>type 1.txt
123
456

E:\TMP>type 2.txt
123
456
```

Рис. Изменение исходного файла

Изменения в файле-ссылке приводят к изменениям в исходном файле, см. рис. На самом деле все изменения происходят только с исходным файлом. Файл-ссылка перенаправляет все запросы к исходному файлу.

```
E:\TMP>echo 789 >> 2.txt

E:\TMP>type 2.txt
123
456
789

E:\TMP>type 1.txt
123
456
789
```

Рис. Изменение файла-ссылки

Если удалить исходный файл, вся информация будет потеряна, см. рис. Файл-ссылка по определению – это только ссылка.

```
E:\TMP>del 1.txt

E:\TMP>dir
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of E:\TMP

09/19/2021  09:52 PM    <DIR>          .
09/19/2021  09:52 PM    <DIR>          ..
09/19/2021  09:37 PM    <SYMLINK>      2.txt [1.txt]
               1 File(s)                0 bytes
               2 Dir(s)  19,210,309,632 bytes free

E:\TMP>type 2.txt
The system cannot find the file specified.
```

Рис. Удаление исходного файла

Задание. Повторите описанные выше опыты.

Задание. Изучите справку для команды **MKLINK**.

BAT/CMD – Пакетные файлы

Пакетный файл операционной системы MS Windows – это файл с расширением **BAT** или **CMD**. Эти три буквы написаны после точки в названии файла, чтобы указать его тип, см. рис.

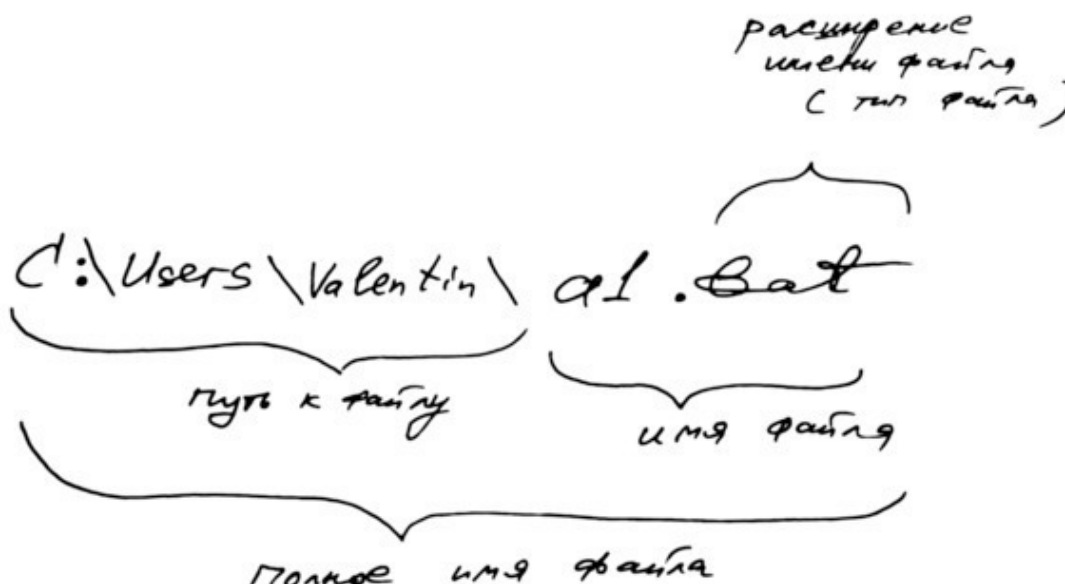


Рис. Имя файла

Напомним, что место расположения файла на диске называется «путь к файлу». Если записать путь к файлу и имя файла, получим «полное имя файла».

С точки зрения ОС это исполняемый файл. Его можно запустить на выполнение.

На самом деле пакетный файл – это текстовый файл. Он содержит команды ОС. Те самые команды, которые мы вводим в командной строке с клавиатуры.

Буквы **BAT** – сокращение английского слова *batch* – «пакет, пакетный». «Пакет» в данном случае – это последовательность из нескольких команд ОС, которая выполняется как один «пакет заданий».

Буквы **CMD** – сокращение английского слова *command* – «команда, командный», как и название программы, обслуживающей командное окно (командную строку).

Другие названия пакетного файла:

- Batch file
- Script
- Файл-сценарий
- Скрипт

Создание и запуск пакетного файла

Мы обсудили самые общие представления. Теперь создадим наш первый пакетный файл.

Шаг 1. Создание пакетного файла – на рисунке выделено красным. Создать пакетный файл можно точно так же, как и любой текстовый файл – с помощью команды:

COPY CON A1.BAT

Введём одну команду **CD** для вывода текущего каталога. Далее нажимаем комбинацию **[Ctrl + Z]** и клавишу **[Enter]**.

Шаг 2. Запуск пакетного файла на выполнение – на рисунке выделено синим. Вводим название файла **A1.BAT** и нажимаем клавишу **[Enter]**. При выполнении файла вна-

чале на экран выводится команда **CD**. В следующей строке выводится результат выполнения команды – текущий каталог.

Шаг 3. Запуск пакетного файла без указания расширения – только те буквы в названии файла, которые идут слева от точки. Вводим **A1** и нажимаем **[Enter]**. Файл выполняется точно так же, как на предыдущем шаге.



```
C:\Users\Valentin>copy con a1.bat
cd
^Z
1 file(s) copied.

C:\Users\Valentin>a1.bat
C:\Users\Valentin>cd
C:\Users\Valentin

C:\Users\Valentin>a1
C:\Users\Valentin>cd
C:\Users\Valentin
```

Рис. Создание и запуск пакетного файла

Задание. Повторите описанные выше шаги.

Задание. Создайте новый пакетный файл, содержащий команду **help /?** и запустите это файл на выполнение двумя способами – с указанием расширения имени файла и без него.

Творческое задание. Просмотрите в Википедии статью **Пакетный файл** и английскую статью **Batch file**.

CMD – Запуск командной строки

В командной строке можно запустить интерпретатор командной строки **CMD.EXE**. Вначале это выглядит странно, потом кое-что прояснится.

Чтобы понять, что происходит, прочитаем описание команды.

Задание. Просмотрите справку для **CMD**. Обратите внимание на ключ **/C**.

Создадим пакетный файл и запустим его на выполнение, как показано на рис. В файле происходит запуск командной строки с выполнение команды. Попробуйте предположить, что произойдёт при выполнении такого пакетного файла.

```
C:\Users\Valentin>copy con x1.bat
cmd /c x1
^Z

    1 file(s) copied.

C:\Users\Valentin>x1

C:\Users\Valentin>cmd /c x1

C:\Users\Valentin>cmd /c x1

C:\Users\Valentin>cmd /c x1
```

Рис. Пакетный файл, запускающий себя

На экране мы увидим что-то не слишком приятное – мелькание одинаковых строк. Это учебное задание, и так к нему и надо относиться. Пользователю надо быть готовым к подобным неприятностям.

Чтобы выйти из этого режима, придётся закрыть командное окно. Можно, например, щёлкнуть правой кнопкой по иконке программы в **Панели задач – Taskbar**. В контекстном меню выбираем пункт **Закреть окно – Close window**, см. рис.

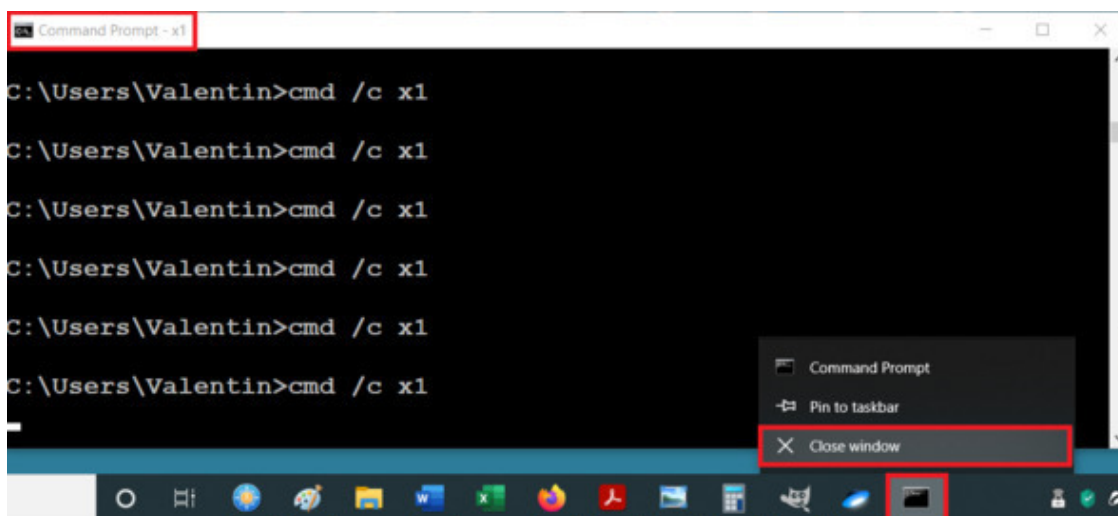


Рис. Аварийный останов программы

Как выясняется, в такой ситуации можно подождать несколько минут и понаблюдать за «переполнением» оперативной памяти в **Диспетчере задач – Task Manager** на вкладке **Память – Memory**, см. рис. Конечно, памяти ещё хватает, но ресурсы ОС уже не позволяют продолжать такой «зацикленный», «вложенный» запуск приложения.

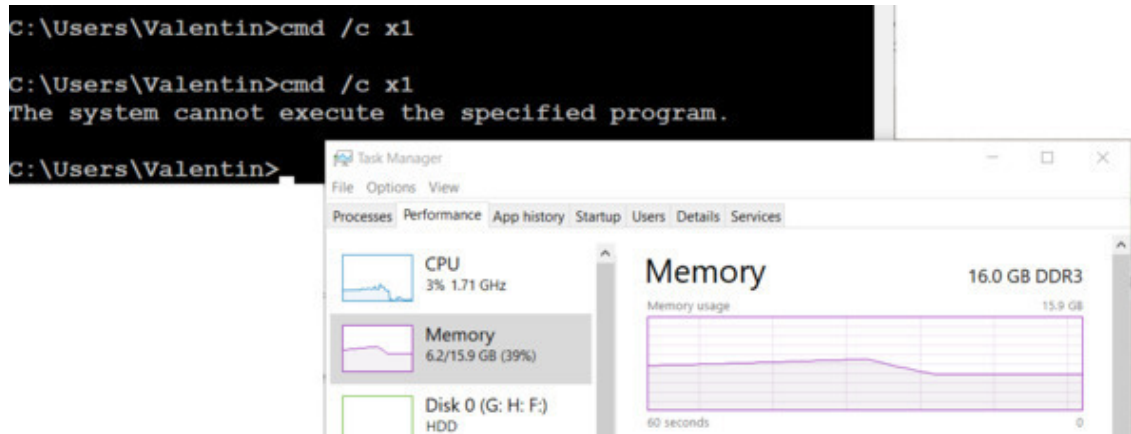


Рис. «Переполнение» памяти

Задание. Повторите описанный опыт.

START – Запуск в новом окне

Командное окно позволяет запустить программу или команду в новом командном окне. Для этого служит команда **START**, см. рис. Как видим, все параметры команды являются необязательными.

```
C:\Users\Valentin>start /?
Starts a separate window to run a specified program or command.

START ["title"] [/D path] [/I] [/MIN] [/MAX] [/SEPARATE | /SHARED]
      [/LOW | /NORMAL | /HIGH | /REALTIME | /ABOVENORMAL | /BELOWNORMAL]
      [/NODE <NUMA node>] [/AFFINITY <hex affinity mask>] [/WAIT] [/B]
      [command/program] [parameters]
```

Рис. Формат команды **START**

Задание. Изучите справку для команды **START**. Предположите, что произойдёт при запуске команды **START** без параметров. Затем запустите команду и проверьте своё предположение.

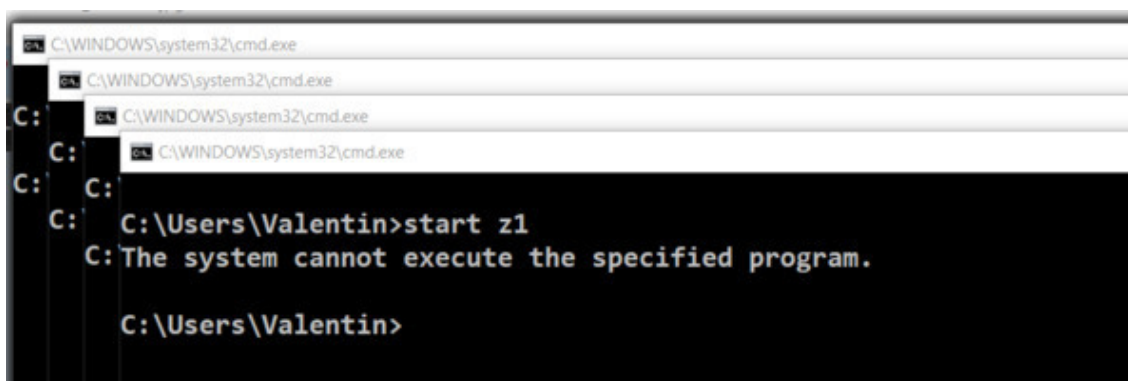
Следующий опыт будет гораздо интереснее. Создадим пакетный файл **Z1.CMD**, содержащий команду **START Z1**. Запустим наш файл на выполнение, см. рис.

```
C:\Users\Valentin>copy con z1.cmd
start z1
^Z
      1 file(s) copied.

C:\Users\Valentin>z1_
```

Рис. Файл **Z1.CMD**

Задание. Создайте пакетный файл **Z1.CMD**, содержащий команду **START Z1**. Что может произойти при запуске такого файла на выполнение? Запустите файл и наблюдайте за происходящим.



```
C:\WINDOWS\system32\cmd.exe
C:\WINDOWS\system32\cmd.exe
C:\WINDOWS\system32\cmd.exe
C:\WINDOWS\system32\cmd.exe
C:\WINDOWS\system32\cmd.exe
C:\Users\Valentin>start z1
C:\The system cannot execute the specified program.

C:\Users\Valentin>
```

Рис. Выполнение пакетного файла **Z1**

Задание. Повторите описанный опыт. Дождитесь сообщения о невозможности запуска. Затем закройте все командные окна через контекстное меню **Панели задач – Close all windows**.

ЕCHO – Вывод сообщения

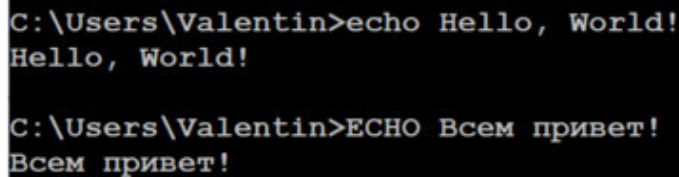
Команда **ЕCHO** выводит текстовое сообщение на экран.

Формат команды очень простой: **ЕCHO текст**. Всё, что идёт после пробела, выводится на экран.

Слово **ЕCHO**, как ни странно, означает «эхо». И читается оно примерно так же.

Задание. Уточните произношение слова **ЕCHO** в любом словаре или переводчике.

Как и прежде, команду можно вводить и заглавными, и строчными буквами. А вот регистр букв в сообщении повлияет на выводимый текст, см. рис.



```
C:\Users\Valentin>echo Hello, World!  
Hello, World!  
  
C:\Users\Valentin>ЕCHO Всем привет!  
Всем привет!
```

Рис. Вывод сообщения

Задание. Выведите на экран своё собственное приветствие с помощью команды **ЕCHO**.

Команда **ЕCHO** может управлять выводом команд пакетного файла во время выполнения. Как мы видели в предыдущих примерах, при выполнении **BAT**-файла на экран выводится команда, а потом результат её выполнения. Эту функцию можно отключить. Тогда на экране мы увидим только результаты работы.

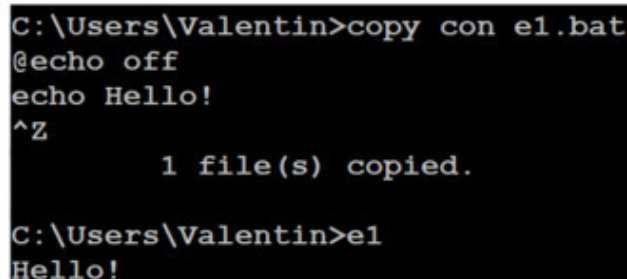
Задание. Изучите справку для команды **ЕCHO**.

Создадим новый пакетный файл, см. рис.

В первой строке у нас будет команда **ЕCHO OFF** – «подавление вывода команды на экран». Обратите внимание, что перед командой стоит значок **@**. Он подавляет вывод текущей команды на экран. Сама команда выполнится, но текст команды на экране не появится.

Во второй строке мы уже выводим наше сообщение на экран.

При выполнении пакетного файла на экране появится одно сообщение – и больше ничего лишнего.



```
C:\Users\Valentin>copy con e1.bat  
@echo off  
echo Hello!  
^Z  
  
1 file(s) copied.  
  
C:\Users\Valentin>e1  
Hello!
```

Рис. Пакетный файл с выводом сообщения

Задание. Повторите описанный опыт.

Задание. Создайте новый пакетный файл и выведите на экран своё собственное сообщение.

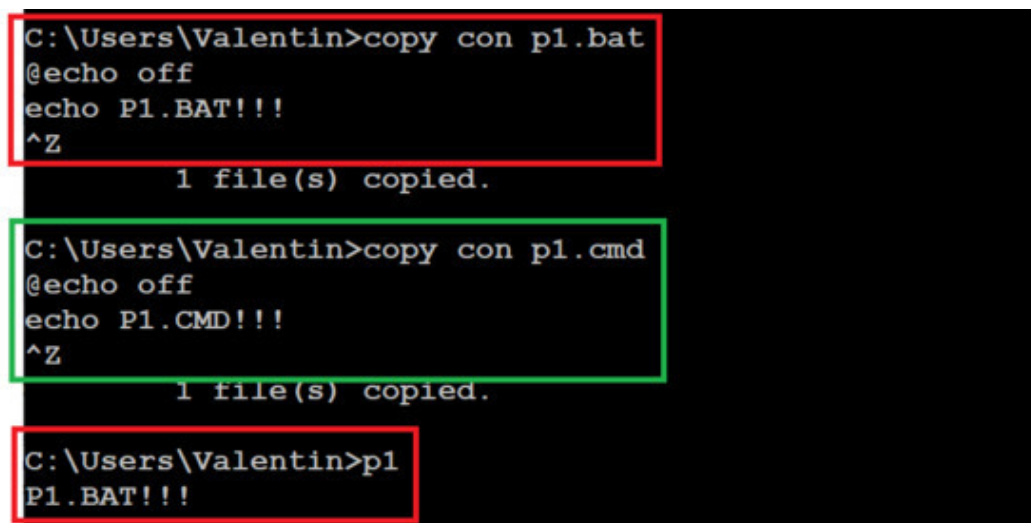
Творческое задание. Просмотрите в Википедии статью @ («собака») и английский вариант статьи **At sign**. Выясните, как этот значок называется, как это название произносится и что он первоначально означал.

Очерёдность вызова

Как мы уже видели, при запуске исполняемого файла не обязательно указывать его расширение. Операционная система сама найдёт нужный файл. Но не всё так очевидно и однозначно.

Пакетные файлы могут иметь расширение **BAT** и **CMD**. Посмотрим, к чему это может привести.

Создадим два пакетных файла с одинаковыми названиями и разными расширениями. Каждый файл выводит на экран своё сообщение. Запускаем на выполнение файл, но не указываем расширение, см. рис.



```
C:\Users\Valentin>copy con p1.bat
@echo off
echo P1.BAT!!!
^Z
1 file(s) copied.

C:\Users\Valentin>copy con p1.cmd
@echo off
echo P1.CMD!!!
^Z
1 file(s) copied.

C:\Users\Valentin>p1
P1.BAT!!!
```

Рис. Два пакетных файла

Задание. Повторите описанный опыт, используя свои собственные сообщения.

Почему был выполнен именно **BAT**-файл, а не **CMD**, мы узнаем в следующем разделе.

% – аргументы командной строки

Пакетный файл тоже может читать параметры (аргументы) командной строки —как и любой исполняемый файл.

Аргументы здесь тоже передают через пробел после имени файла. К этим аргументам можно обращаться внутри пакетного файла следующим образом:

%0 – нулевой аргумент (имя команды)

%1 – первый аргумент

...

%9 – девятый аргумент.

Создадим простой пакетный файл, который выводит на экран нулей и первый аргументы командной строки, см. рис.

```

E:\TMP>copy con par1.cmd
@echo off
echo %0
echo %1
^Z

        1 file(s) copied.

E:\TMP>par1 1
par1
1

E:\TMP>par1 one
par1
one

E:\TMP>par1
par1
one
ECHO is off.
  
```

Рис. Два аргумента

Задание. Повторите описанный опыт.

Задание. Создайте пакетный файл, который выводит первый и третий аргументы на экран.

Задание. Проверьте, как пакетный файл воспринимает запятую в середине текста аргумента командной строки.

Обратим внимание на нулевой аргумент. Он сообщает, как именно мы вызывали наш пакетный файл – с расширением или без, с указанием каталога или без. Между прочим, здесь тоже можно использовать точку для обозначения текущего каталога, см. рис.

```

E:\TMP>par1
par1
ECHO is off.

E:\TMP>.\par1
.\par1
ECHO is off.

E:\TMP>par1.cmd 2
par1.cmd
2

E:\TMP>e:\tmp\par1 next
e:\tmp\par1
next
  
```

Рис. Нулевой аргумент

Точка и обратная косая черта указывают на текущий каталог.

Если пакетный файл пытается обратиться к аргументу, которого нет, выводится следующее сообщение:

ECHO is off.

И это неспроста.

Попробуем разобраться, что же здесь происходит.

Создадим новый пакетный файл.

Будем выводить на экран параметры **%0**, **%1** и **%2**.

Затем вызываем команду **ECHO** без параметров.

Как видим, при отсутствии аргумента для вывода наша команда **ECHO** ведёт себя так, как будто у неё нет параметров, см. рис.

```

E:\TMP>copy con par2.cmd
@echo off
echo %0
echo %1
echo %2
echo
^Z
1 file(s) copied.

E:\TMP>par2
par2
ECHO is off.
ECHO is off.
ECHO is off.

E:\TMP>par2 one
par2
one
ECHO is off.
ECHO is off.

E:\TMP>par2 one two
par2
one
two
ECHO is off.

```

Рис. Вывод параметров командой **ECHO**

Вызываем справку для команды **ECHO**, см. рис. Выясняем, что команда **ECHO** без параметров выводит на экран свои текущие настройки. А эти настройки включают в себя всего два варианта:

ECHO ON

ECHO OFF

```

E:\TMP>echo /?
Displays messages, or turns command-echoing on or off.

ECHO [ON | OFF]
ECHO [message]

Type ECHO without parameters to display the current echo setting.

```

Рис. Справка для команды **ECHO**

Внутри пакетного файла в первой строке мы как раз и отключаем режим вывода команд на экран:

echo off

Поэтому при следующем вызове команды **ECHO** нам сообщают, что вывод команд отключён:

Echo is off.

В одном из примеров мы пытаемся вызвать команду **ECHO** и передать её несуществующий второй параметр командной строки:

echo %2

Получается, что это равносильно вызову команды **ECHO** без параметров. Вот она нам и сообщает о своей настройке – в соответствии с описанием команды.

Задание. Повторите описанный опыт.

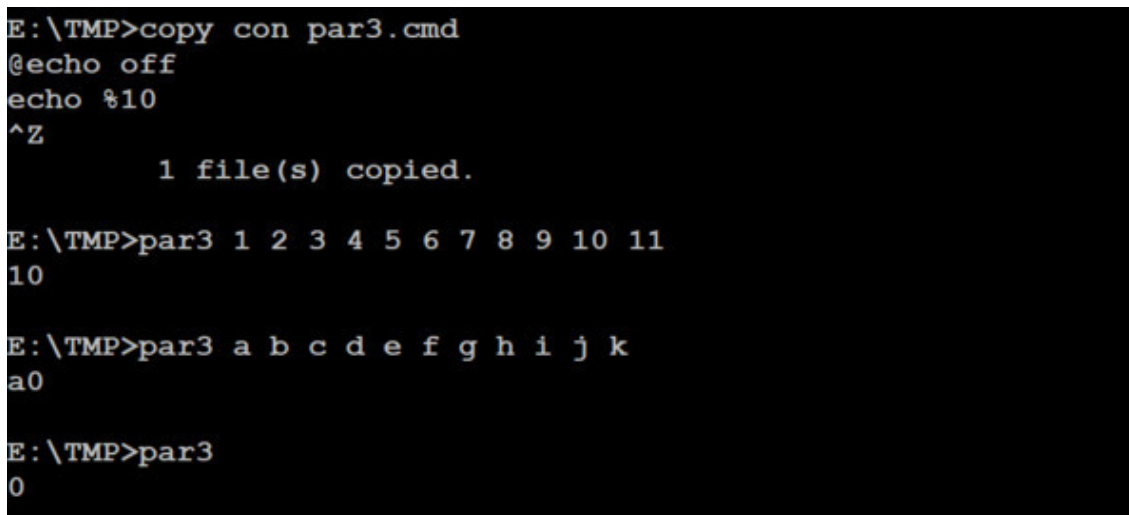
Творческое задание. Отключите режим вывода команд **ECHO OFF** в командной строке (не внутри пакетного файла). Затем введите пару команд ОС и обратите внимание на то, что происходит на экране. Затем включите вывод команд: **ECHO ON** и введите пару команд.

Знак процента позволяет обратиться к параметрам с номерами от 0 до 9: от **%0** до **%9**.

Попробуем вызвать команду:

ECHO %10

Составим простой пакетный файл и дадим ему на вход не меньше десяти параметров, см. рис.



```

E:\TMP>copy con par3.cmd
@echo off
echo %10
^Z
        1 file(s) copied.

E:\TMP>par3 1 2 3 4 5 6 7 8 9 10 11
10

E:\TMP>par3 a b c d e f g h i j k
a0

E:\TMP>par3
0
  
```

Рис. Десятый параметр

При первом вызове мы получаем на экране **10**. Вроде бы правильно.

При втором вызове вводим буквы в качестве аргументов. Получаем **a0**.

Наконец, вызываем пакетный файл без параметров и получаем **0**.

Получается, что команда **ECHO %10** работает, как команда **ECHO %1**, а затем печатает ноль.

Если же десятого аргумента нет, то команда работает как **ECHO 0**.

Мы убедились, что с помощью значка «процент» дальше девятого аргумента нельзя «дотянуться».

Задание. Повторите описанный опыт.

FOR – Цикл

Пакетный файл – это программа. Здесь каждая строчка – это команда ОС. Кроме обычных команд, в пакетных файлах есть средства программирования. В том числе есть оператор цикла **FOR**.

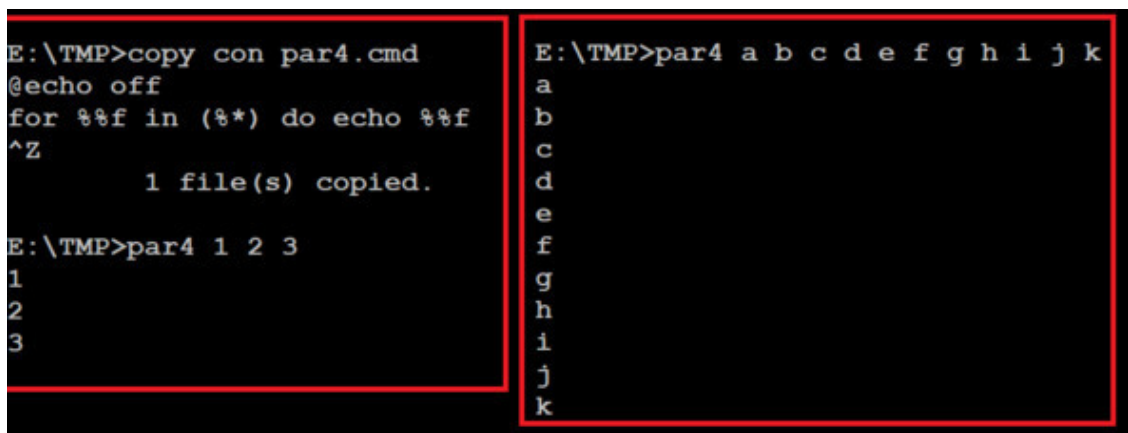
Создадим пакетный файл и организуем в цикле вывод всех аргументов командной строки:

```
@echo off
```

```
for %%f in (*) do echo %%f
```

Здесь `%%f` – это переменная, которая последовательно принимает значения из списка `(*)` – то есть из списка аргументов командной строки.

Проверим, как сработает такой подход, см. рис.



```
E:\TMP>copy con par4.cmd
@echo off
for %%f in (*) do echo %%f
^Z
1 file(s) copied.

E:\TMP>par4 1 2 3
1
2
3
```

```
E:\TMP>par4 a b c d e f g h i j k
a
b
c
d
e
f
g
h
i
j
k
```

Рис. Вывод аргументов в цикле

Задание. Повторите описанный опыт.

Творческое задание. Изучите справку для оператора **FOR**. Составьте пакетный файл, который выводит на экран содержимое всех текстовых файлов в текущем каталоге.

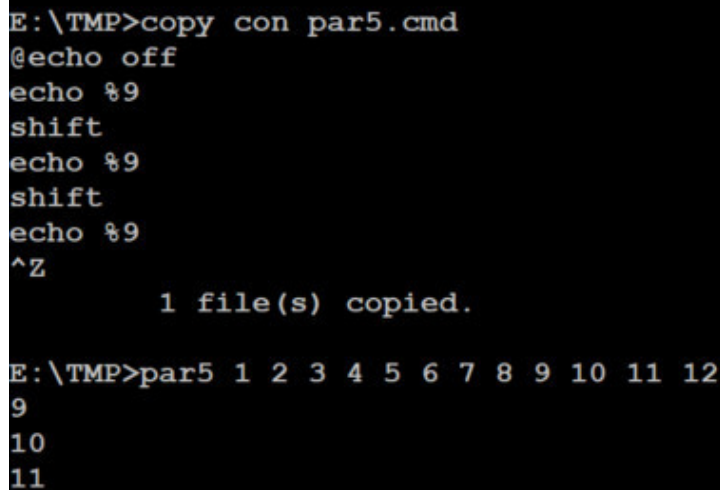
SHIFT – смещение нумерации аргументов

В пакетных файлах есть средство, чтобы всё-таки добраться до десятого аргумента. И это команда **SHIFT** – сдвиг нумерации аргументов.

Если вызвать команду **SHIFT**, нумерация «сдвигается» на единицу:

%1 будет обращаться ко второму аргументу

%9 обратится к десятому аргументу, см. рис.



```
E:\TMP>copy con par5.cmd
@echo off
echo %9
shift
echo %9
shift
echo %9
^Z
          1 file(s) copied.

E:\TMP>par5 1 2 3 4 5 6 7 8 9 10 11 12
9
10
11
```

Рис. Сдвиг нумерации

Таким образом, чтобы добраться до аргумента с номером **11**, придётся сдвинуть нумерацию два раза. Тогда можно будет обратиться к **11**-му элементу с помощью ссылки **%9**.

Задание. Повторите описанный опыт.

Творческое задание. Изучите справку для команды **SHIFT**. Изучите справку для команды **SHIFT**. Выясните, что делает команда **SHIFT /5** и проверьте своё предположение.

SET – Параметры окружения

В операционных системах имеются особые вспомогательные «переменные», которые находятся в оперативной памяти. Это текстовые строки, к которым обращаются по имени – почти как к переменным в программировании. Только в данном случае к параметрам окружения могут обращаться прикладные программы и команды ОС. С помощью этих параметров можно управлять поведением программ и выполнением команд.

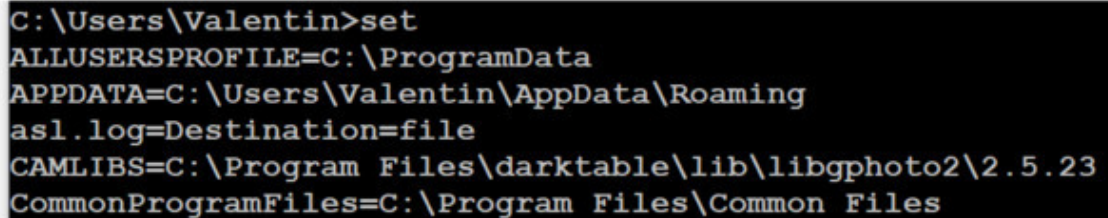
Названия этих переменных могут быть самые разнообразные:

- Переменные окружения
- Параметры окружения
- Переменные среды
- Environment variables
- SET-параметры.

SET – Вывод параметров

Список параметров окружения и их значения можно вывести на экран. Для этого служит команда **SET**.

Вызываем команду **SET** без параметров и получаем довольно длинный список, см. рис.

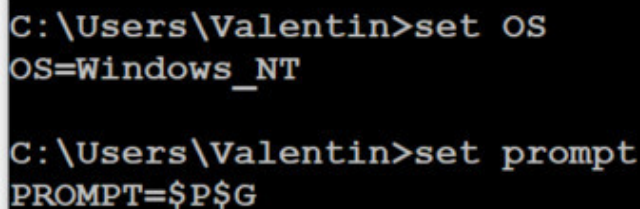


```
C:\Users\Valentin>set
ALLUSERSPROFILE=C:\ProgramData
APPDATA=C:\Users\Valentin\AppData\Roaming
asl.log=Destination=file
CAMLIBS=C:\Program Files\darktable\lib\libgphoto2\2.5.23
CommonProgramFiles=C:\Program Files\Common Files
```

Рис. Параметры окружения

В начале каждой строки видим название параметра (переменной), затем знак «равно» и значение параметра.

Чтобы вывести на экран значение одного выбранного параметра, вызываем команду **SET** и указываем название параметра. Можно, например, вывести название операционной системы или узнать настройку приглашения к вводу команд, см. рис.



```
C:\Users\Valentin>set OS
OS=Windows_NT

C:\Users\Valentin>set prompt
PROMPT=$P$G
```

Рис. Вывод одного параметра

В дальнейших опытах нас будут интересовать два параметра:

- **Path** – путь для поиска исполняемых файлов;
- **PATHEXT** – расширения имен исполняемых файлов.

Слово **Path** означает «путь», а точнее «путь к файлу».

Сокращение **PATHEXT** образовано из слов **Path** – «путь» и **Extension** – «расширение имени файла».

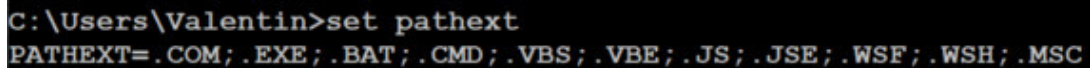
Задание. Выведите на экран значения переменных **Path** и **PATHEXT**. Попробуйте использовать разный регистр при вводе имени переменной окружения.

Задание. Просмотрите в Википедии статью **Путь к файлу** и английский вариант статьи **Path (computing)**.

Задание. Просмотрите в Википедии раздел **Execution preference** английской статьи **COM file**.

PATHEXT – Порядок запуска

Переменная **PATHEXT** перечисляет расширения файлов. В соответствии с этим списком ОС выбирает программу для запуска, если есть несколько одноимённых файлов с разными расширениями см. рис.



```
C:\Users\Valentin>set pathext
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
```

Рис. Переменная **PATHEXT**

Напомним, что в предыдущем опыте мы создали два пакетных файла **P1.BAT** и **P1.CMD** в одном каталоге. Затем мы запустили программу **P1** без указания расширения.

В переменной **PATHEXT** сначала идёт расширение **BAT**, затем **CMD**. В соответствии с этим порядком, ОС выбирает для запуска **P1.BAT**. При этом второй файл **P1.CMD** не будет запущен.

PATH – Путь для поиска

Переменная **Path** указывает операционной системе, где искать программу для запуска – если мы не указали каталог.

Обычно поиск начинается в текущем каталоге текущего диска. Если здесь не удалось найти подходящий файл, то ОС продолжит поиск по каталогам, перечисленным в переменной **Path**. Будет запущена первая найденная программа. Но при этом продолжает действовать порядок запуска, описанный в переменной **PATHEXT**. Если в одном каталоге есть несколько «одноимённых» программ с разным расширением, используется заданная в **PATHEXT** очередность.

Проведём опыт с запуском файла из другого каталога. В текущем каталоге уже есть два пакетных файла **P1**. Перейдём на другой диск. Запустим программу **P1** на выполнение, не указывая путь и расширение, см. рис.

```
C:\Users\Valentin>dir /b p1.*
p1.bat
p1.cmd

C:\Users\Valentin>p1
P1.BAT!!!

C:\Users\Valentin>d:

D:\>p1
'p1' is not recognized as an internal or external command,
operable program or batch file.
```

Рис. Запуск из другого каталога

Продолжим наш опыт.

Чтобы ОС могла найти наш пакетный файл, добавим домашний каталог в путь для поиска.

Вначале выводим на экран значение переменной **Path**:

set path

Как и раньше, можем использовать команды и названия в любом регистре (это сработает только в данной операционной системе).

Как видим, в списке **Path** пути для поиска разделяются точкой с запятой. В нашем примере оказалось, что в конце списка уже есть точка с запятой.

Нам осталось только добавить свой каталог в путь для поиска:

set path=%path%c:\users\valentin

Здесь выражение **%path%** даёт нам значение указанной переменной окружения.

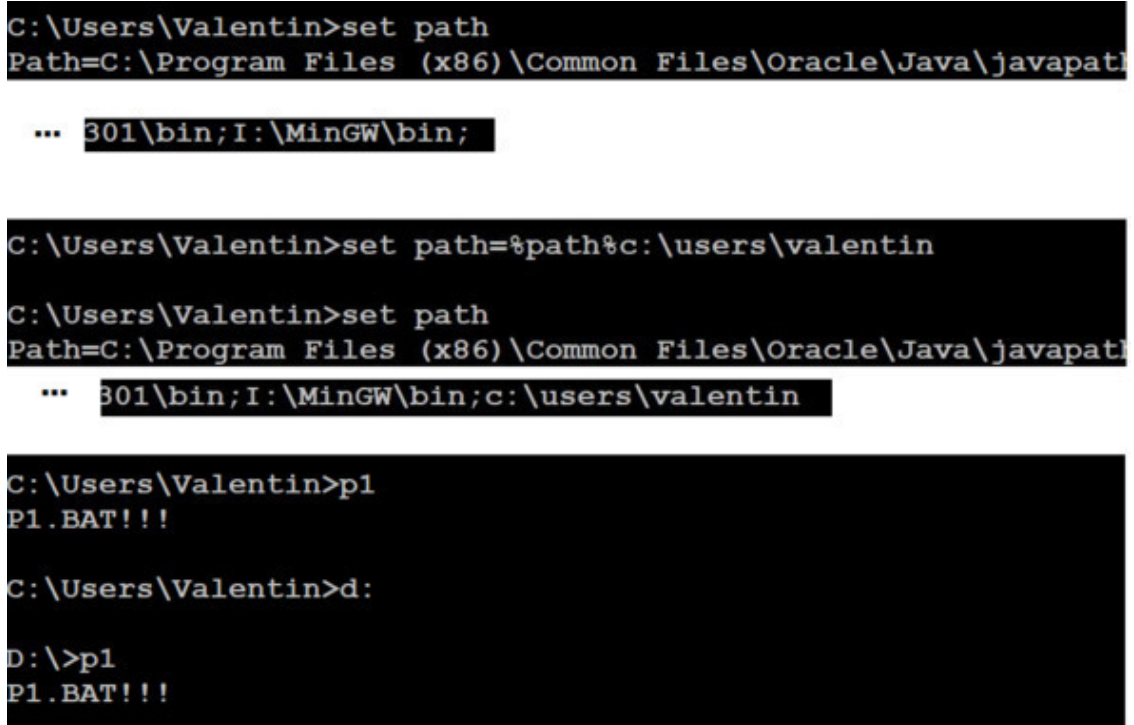
Таким образом, мы сохраняем предыдущий список и дописываем к нему в конце свой каталог.

Далее проверяем содержимое переменной **Path**:

set path

Теперь мы можем запускать свой пакетный файл из любого каталога. ОС найдёт его в соответствии с переменной **Path**.

Но поскольку у нас два файла **P1**, будет запущен именно **P1.BAT**, потому что расширение **BAT** указано в списке **PATHEXT** раньше, чем **CMD**, см. рис.



```
C:\Users\Valentin>set path
Path=C:\Program Files (x86)\Common Files\Oracle\Java\javapath
... 301\bin;I:\MinGW\bin;

C:\Users\Valentin>set path=%path%c:\users\valentin
C:\Users\Valentin>set path
Path=C:\Program Files (x86)\Common Files\Oracle\Java\javapath
... 301\bin;I:\MinGW\bin;c:\users\valentin

C:\Users\Valentin>p1
P1.BAT!!!

C:\Users\Valentin>d:
D:\>p1
P1.BAT!!!
```

Рис. Запуск с поиском по пути

Задание. Повторите описанный опыт.

Наша настройка пути для поиска сохраняется только для текущего командного окна и только на текущий сеанс работы. В другом командном окне сохраняются общие настройки.

Задание. Запустите новое командное окно. Перейдите на другой диск и запустите программу **P1**. Прочитайте сообщение об ошибке.

С переменной **Path** можно работать с помощью команды **PATH**.

Задание. Изучите справку для команды **PATH**.

Задание. Запустите команду **PATH** без параметров.

Творческое задание. Просмотрите в Википедии статью **Путь к файлу** и английскую статью **Path (computing)**.

7-ZIP – архиватор файлов

Программа 7-ZIP – это пример бесплатного архиватора.

Архиватор позволяет упаковывать файлы в один архив, в том числе с сохранением структуры каталогов.

Задание. Просмотрите в Википедии статью **7-Zip**.

На самом деле есть разница между бесплатным и свободным программным обеспечением. А ещё существует программное обеспечение с открытым исходным кодом. Для пользователя часто бывает достаточно выяснить, программа бесплатная. Но попробуем немного разобраться с этим вопросом. Выясняется, что есть целых три разные категории «бесплатного софта» и даже их комбинации.

Задание. Просмотрите в Википедии статью **Архиватор**.

Творческое задание. Просмотрите в Википедии следующие статьи

Свободное программное обеспечение

Free software

Бесплатное программное обеспечение

Freeware

Открытое программное обеспечение

Open-source software

Свободное и открытое программное обеспечение

Free and open-source software

Выясните, в чём разница между Free software, Freeware и Open-source software.

Загрузка программы

Адрес для скачивания нашего архиватора:

<https://www.7-zip.org/>

Задание. Перейдите на указанный сайт и выясните, к какой категории ПО относится данная программа.

В разделе **Download** находим несколько вариантов для загрузки:

- **.EXE** – программа установки
- **.MSI** – альтернативный установщик
- **.7Z** – консольная версия

Здесь также есть возможность скачать исходный текст программы – **Source code**, см. рис.

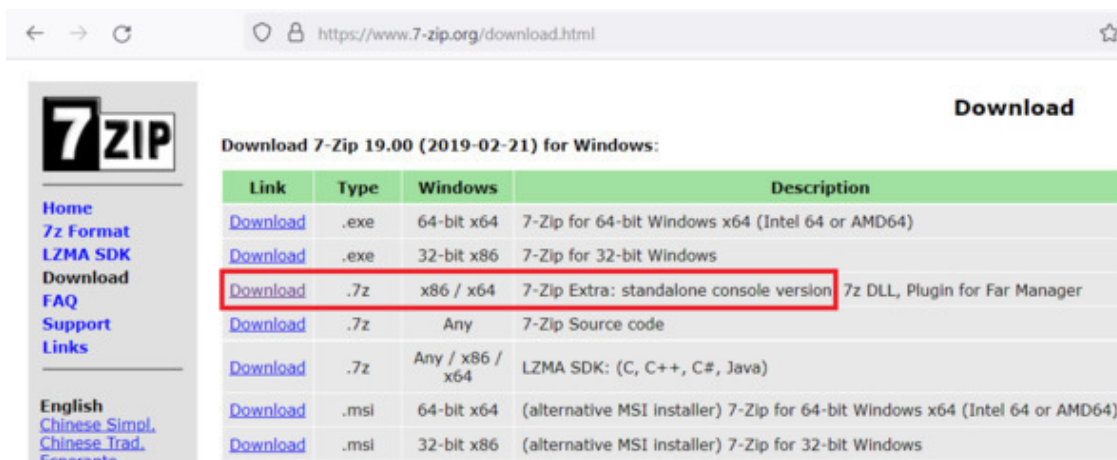


Рис. Варианты загрузки

Для установки программы могут потребоваться права администратора.

В наших опытах попробуем использовать консольную (переносимую) версию. Этот вариант программы будет работать без установки и только в командной строке.

Правда, есть одно препятствие: мы скачиваем программу, упакованную в архив. Чтобы её использовать, нужно распаковать (разархивировать) файл ***.7z**. Это для вас будет творческим заданием.

Дальше работа пойдёт по плану.

Мы размещаем полученные файлы в каком-нибудь каталоге. Нам понадобятся всего три файла, название которых начинается на **7z**, см. рис.

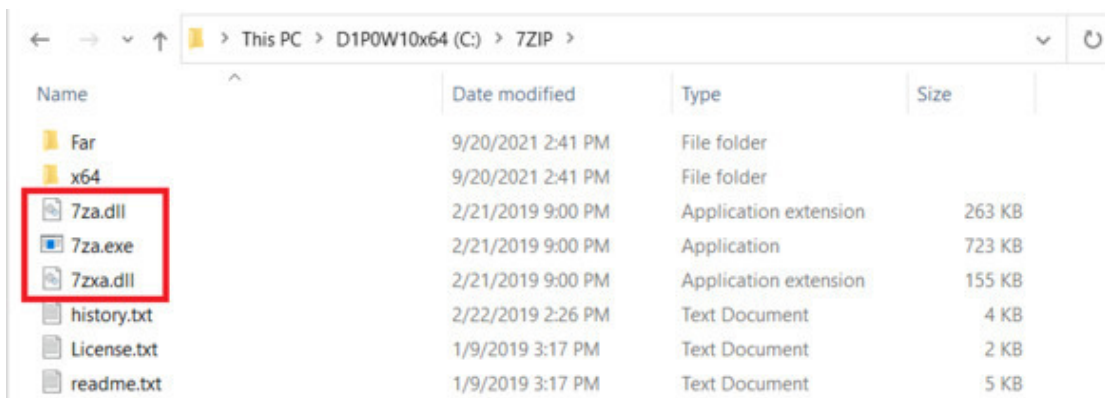


Рис. Расположение файлов архиватора

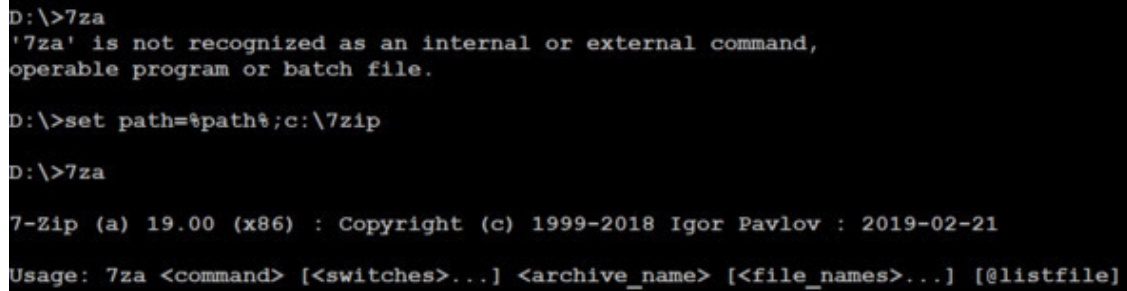
В нашем примере исполняемые файлы архиватора находятся в каталоге **C:\7ZIP**. В вашем случае это может быть другой диск и другой каталог – куда есть доступ для пользователя.

Задание. Скачайте консольную версию архиватора и расположите её в удобном месте.

PATH – Путь к программе

Чтобы использовать архиватор, понадобится установить путь для поиска, то есть переменную **Path**. Мы с этим уже познакомились.

Добавляем путь к каталогу архиватора в путь для поиска. Убеждаемся, что программу можно вызывать и она нам что-то сообщает, см. рис.



```
D:\>7za
'7za' is not recognized as an internal or external command,
operable program or batch file.

D:\>set path=%path%;c:\7zip

D:\>7za

7-Zip (a) 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21
Usage: 7za <command> [<switches>...] <archive_name> [<file_names>...] [@listfile]
```

Рис. Установка пути для поиска

Как видим, при вызове архиватора есть три обязательных элемента: имя программы, команда и имя архива.

Задание. Установите путь поиска для архиватора. Убедитесь, что программа работает.

ADD – Создание архива

Начнём с самого простого действия – взять файл и поместить его в архив. Для этого нам понадобится команда **a** – *Add files to archive*. Буква **a** – первая буква слова *Add* – «добавить».

Для опытов скачаем какое-нибудь изображение. На сайте **Картинки Яндекса** можно найти много интересного и приятного. Выбираем первую попавшуюся картинку из категории «Ледник», см. рис.

Выбираем лучшее качество изображения. В нашем примере это картинка с размерами 1600 на 1066 точек.

Сохраняем файл в выбранном каталоге.

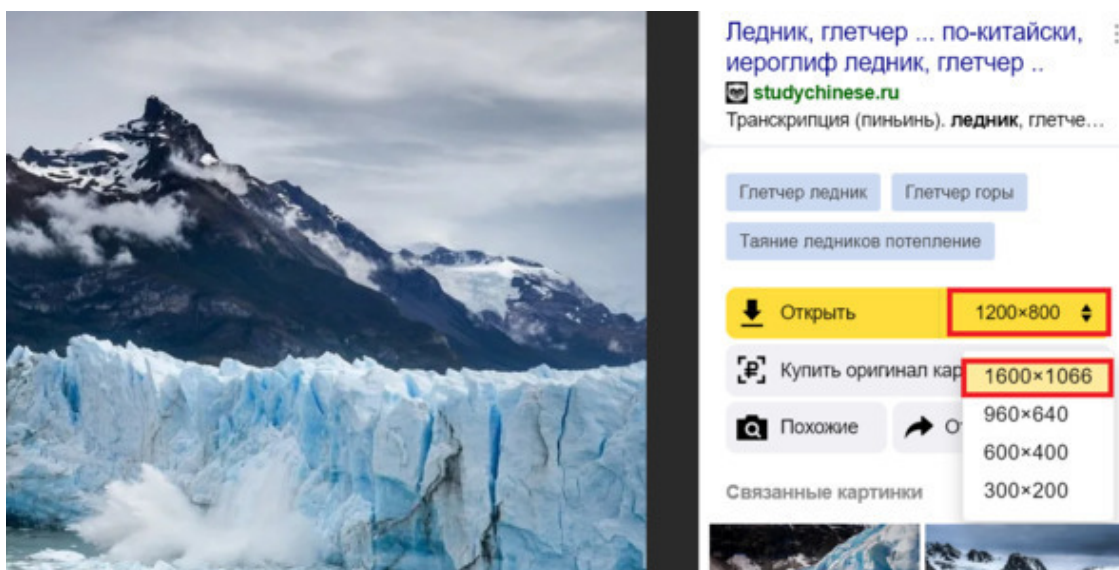


Рис. Выбор изображения

Переходим в нужный каталог и выясняем, каких размеров файл мы получили. В нашем примере файл имеет размеры около 400 килобайт, см. рис.

```
D:\>e:

E:\>cd tmp

E:\TMP>dir /b
dsc04348.jpg

E:\TMP>dir
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of E:\TMP

09/20/2021  03:54 PM    <DIR>          .
09/20/2021  03:54 PM    <DIR>          ..
09/20/2021  03:51 PM             418,084 dsc04348.jpg
               1 File(s)              418,084 bytes
               2 Dir(s)  19,212,550,144 bytes free
```

Рис. Размеры файла

Для удобства дальнейшей работы переименуем наш файл и дадим ему короткое имя **x1.jpg**:

```
ren dsc04348.jpg x1.*
```

В данном случае **ren** – это команда переименования **RENAME**.

Новое имя файла **x1**, а расширение оставим прежнее. Для этого вместо расширения поставим символ «звёздочку».

Убедимся, что в нашем каталоге имеется файл под новым названием, но прежнего размера, см. рис.

```
E:\TMP>ren dsc04348.jpg x1.*

E:\TMP>dir
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of E:\TMP

09/20/2021  03:57 PM    <DIR>          .
09/20/2021  03:57 PM    <DIR>          ..
09/20/2021  03:51 PM             418,084 x1.jpg
                   1 File(s)                418,084 bytes
                   2 Dir(s)  19,212,513,280 bytes free
```

Рис. Переименование файла

Переходим к архивированию. Вызываем следующую команду:

```
7za a arch1 x1.jpg
```

Здесь мы указали имя программы-архиватора **7za** и не сообщили, где она находится. За счёт настройки переменной окружения **Path** мы можем вызывать архиватор из любого каталога.

Команда **a** – добавить файл в архив.

Название файла **arch1** – имя будущего архива.

Файл **x1.jpg** – наш объект для упаковки (архивации).

```
E:\TMP>7za a arch1 x1.jpg

7-Zip (a) 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov

Scanning the drive:
1 file, 418084 bytes (409 KiB)

Creating archive: arch1.7z

Add new data to archive: 1 file, 418084 bytes (409 KiB)

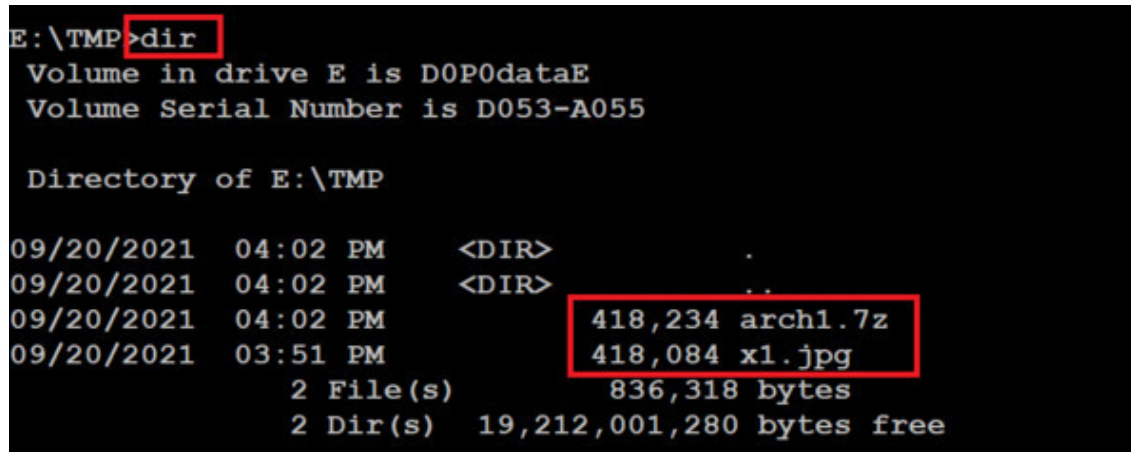
Files read from disk: 1
Archive size: 418234 bytes (409 KiB)
Everything is Ok
```

Рис. Создание архива

Обратим внимание, что размеры файла указаны в байтах и в загадочных KiB.

Задание. Просмотрите в Википедии статью **Байт**. Выясните, как называется новая единица измерения **KiB** и чем она отличается от килобайта.

С помощью команды **DIR** выясняем, что произошло с размерами файла, см. рис.



```
E:\TMP>dir
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of E:\TMP

09/20/2021  04:02 PM    <DIR>          .
09/20/2021  04:02 PM    <DIR>          ..
09/20/2021  04:02 PM                418,234 arch1.7z
09/20/2021  03:51 PM                418,084 x1.jpg
                2 File(s)                836,318 bytes
                2 Dir(s)  19,212,001,280 bytes free
```

Рис. Размеры файлов

Задание. Заархивируйте свой файл и выясните, как изменились размеры.

Задание. Скачайте ещё несколько файлов изображений и добавьте их в общий архив. Сравните суммарный размер исходных файлов и размер архива.

LIST – Оглавление архива

Следующий шаг – получить список файлов, находящихся в архиве. Другими словами, вывести на экран оглавление архива.

Нам понадобится команда **l** – от слова *List* – «список, составить список»:

7za l arch1.7z

Здесь мы указываем не только имя архива, но и его расширение.

```
E:\TMP>7za l arch1.7z

7-Zip (a) 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov :

Scanning the drive for archives:
1 file, 418234 bytes (409 KiB)

Listing archive: arch1.7z

--
Path = arch1.7z
Type = 7z
Physical Size = 418234
Headers Size = 122
Method = LZMA2:19
Solid = -
Blocks = 1
```

Date	Time	Attr	Size	Compressed	Name
2021-09-20	15:51:01	A	418084	418112	x1.jpg
2021-09-20	15:51:01		418084	418112	1 files

Рис. Оглавление архива

Обратим внимание на сведения об атрибутах файла – в колонке **Attr**.

Кроме того, архиватор сообщает, что использует метод LZMA для упаковки файлов.

Задание. Выведите на экран оглавление своего архива. Сравните указанные размеры файлов с результатами работы команды **DIR**.

Творческое задание. Просмотрите в Википедии статью **LZMA**.

TZIP – Создание ZIP-архива

Как мы уже убедились, по умолчанию программа создаёт архив в своём формате **7z**.

В некоторых случаях требуется создать архив в другом, более распространённом, «стандартном» формате **ZIP**.

В этом случае при создании архива нужно указать тип с помощью опции (ключа) **-t** – type – тип архива:

7za a -tzip arch2 x1.jpg.

Здесь мы не указываем расширение имени архива. На выходе получаем файл ***.zip**, см рис.

```
E:\TMP>7za a -tzip arch2 x1.jpg

7-Zip (a) 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov :
Scanning the drive:
1 file, 418084 bytes (409 KiB)

Creating archive: arch2.zip

Add new data to archive: 1 file, 418084 bytes (409 KiB)

Files read from disk: 1
Archive size: 418116 bytes (409 KiB)
Everything is Ok

E:\TMP>dir
Volume in drive E is D0P0dataE
Volume Serial Number is D053-A055

Directory of E:\TMP

09/20/2021  04:46 PM    <DIR>          .
09/20/2021  04:46 PM    <DIR>          ..
09/20/2021  04:02 PM             418,234 arch1.7z
09/20/2021  04:46 PM             418,116 arch2.zip
09/20/2021  03:51 PM             418,084 x1.jpg
               3 File(s)              1,254,434 bytes
               2 Dir(s)  19,211,104,256 bytes free
```

Рис. Создание ZIP-архива

Задание. Упакуйте свои файлы в ZIP-архив.

Задание. Выведите оглавление архива на экран.

Задание. Просмотрите в Википедии статью ZIP.

SFX – самораспаковывающийся архив

Обычно для распаковки архива требуется программа-архиватор. Однако, можно создать архив, который сам может развернуть, распаковать, разархивировать, извлечь своё содержимое. Это так называемый самораспаковывающийся архив. Английское название: Self-Extracting Archive – «архив, который сам извлекает своё содержимое». Сокращённо **SFX**. Это сокращение построено ПОЧТИ по первым буквам слов в названии: Self-eXtracting.

Эта функция так и называется **SFX**, причём без всяких объяснений и даже без расшифровки, см. рис.



```
C:\Users\Valentin>7za

7-Zip (a) 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov : 2019-02-21

Usage: 7za <command> [<switches>...] <archive_name> [<file_names>...] [@listfile]

<Commands>
a : Add files to archive
...
<Switches>
...
-sfx[{name}] : Create SFX archive
```

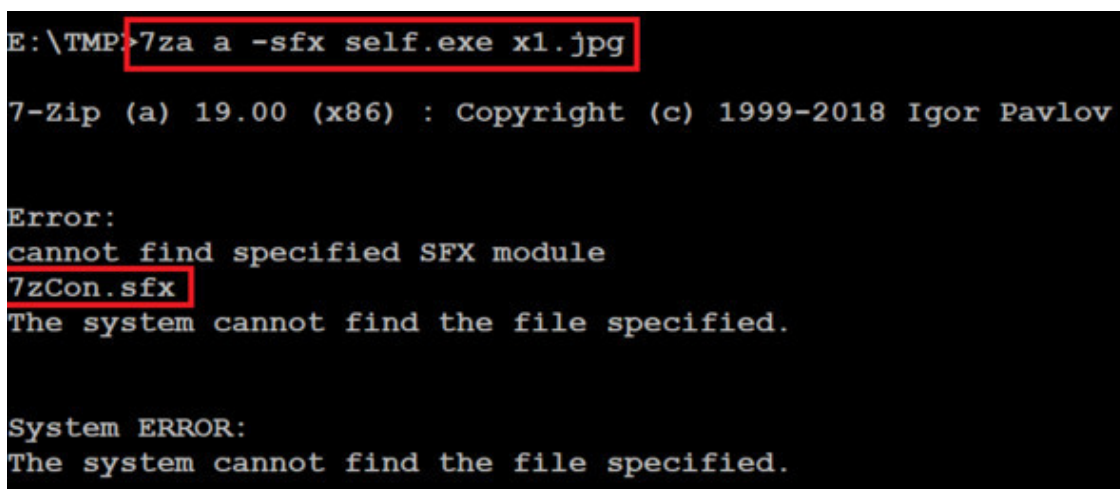
Рис. Описание функции **SFX**

Пользователь должен это знать заранее. Это вроде как само собой разумеется. Но мы на всякий случай это объясним.

Файл формата **SFX** – это исполняемый файл. В ОС Windows он имеет расширение **EXE** – от слова Executable – «исполняемый». Внутри такого файла находится программа для разархивирования и содержимое архива.

Судя по описанию, нам нужна команда **a** для создания архива и ключ **-sfx** для указания формата самораспаковывающегося архива.

Вызываем команду и читаем сообщение на экране, см. рис.



```
E:\TMP>7za a -sfx self.exe x1.jpg

7-Zip (a) 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov

Error:
cannot find specified SFX module
7zCon.sfx
The system cannot find the file specified.

System ERROR:
The system cannot find the file specified.
```

Рис. Требуется файл

Оказывается, нам потребуется дополнительный файл для консольной работы (то есть в командной строке):

7zCon. sfx

Поиск в интернете такую строчку:

7zCon. sfx download

Попробуем скачать найденный файл в текущий каталог.

Снова запустим команду создания архива. На этот раз всё получается, см. рис.

```
E:\TMP>7za a -sfx self.exe x1.jpg

7-Zip (a) 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov

Scanning the drive:
1 file, 418084 bytes (409 KiB)

Creating archive: self.exe

Add new data to archive: 1 file, 418084 bytes (409 KiB)

Write SFX: 7zCon.sfx : 186880 bytes (183 KiB)

Files read from disk: 1
Archive size: 605114 bytes (591 KiB)
Everything is Ok
```

Рис. Создание самораспаковывающегося архива

Содержимое созданного архива можно посмотреть командой **l – LIST**.

Конечно же, такой файл можно запустить на выполнение. Достаточно просто ввести название файла. Расширение указывать необязательно.

Задание. Запустите команду создания самораспаковывающегося архива и прочитайте сообщение об ошибке.

Задание. Скачайте недостающий файл.

Задание. Создайте самораспаковывающийся архив.

Задание. Удалите исходный файл командой **del**.

Задание. Запустите свой **EXE**-файл.

Задание. Убедитесь, что исходный файл появился в текущем каталоге – с помощью команды **DIR**.

Задание. Запустите свой **EXE**-файл снова и подтвердите перезапись файла, ответив **y (YES)**.

Творческое задание. Просмотрите в Википедии статью **Самораспаковывающийся архив** и английскую статью **Self-Extracting Archive**.

По умолчанию самораспаковывающийся архив разворачивается в текущем каталоге. Мы можем сами указать определённый каталог для распаковки.

Для этого используем дополнительный параметр командной строки – ключ **-o**. Буква **o** намекает на английское слово Output – «вывод», то есть «куда именно выводить результаты» или «куда отправить выходной файл». Название каталога указываем сразу после ключа в кавычках, см. рис.

```
E:\TMP>self -o".\1"

7-Zip SFX 19.00 (x86) : Copyright (c) 1999-2018 Igor Pavlov

Extracting archive: E:\TMP\self.exe
--
Path = E:\TMP\self.exe
Type = 7z

Everything is Ok
```

Рис. Распаковка в указанный каталог

В нашем примере точка означает ТЕКУЩИЙ КАТАЛОГ. Это секретное обозначение широко используется в ОС UNIX и Linux. Как видим, эта самая точка иногда встречается и в ОС MS Windows.

Задание. Запустите свой **EXE**-файл и укажите каталог для распаковки, как показано на рисунке.

Задание. Перейдите в указанный подкаталог командой **CD**.

Задание. Убедитесь, что в новом подкаталоге появился ваш файл – с помощью команды **DIR**.

Творческое задание. Изучите справку для **7za** и разберитесь, как создать многотомный архив – обычный и самораспаковывающийся. Поэкспериментируйте с размерами тома.

Творческое задание. Изучите справку для **7za** и разберитесь, как создать архив с сохранением структуры подкаталогов. Поэкспериментируйте с деревом каталогов.

Заключение

Мы рассмотрели основные команды операционной системы **Microsoft Windows**, которые можно вводить в командной строке.

Команд много, и все их запоминать не обязательно. Всегда можно вызвать **HELP** и посмотреть список команд. Описание любой команды нам тоже выведет команда **HELP**.

Ещё раз скажем: это были только **ОСНОВНЫЕ** команды. И это были только **НЕКОТОРЫЕ** примеры работы с этими несколькими командами.

Здесь ещё есть, что освоить и чему научиться. Конечно, если будет такое желание и такая необходимость по работе.

Мы ещё вернёмся к работе в командной строке, изучая ОС Linux, и сразу же заметим много общего. Даже названия многих команд будут знакомы. Ведь любая командная строка – это просто строка для ввода команд :-)

Литература

1. Мажарцев Д. Руководство по командной строке Windows:
<https://cmd.readthedocs.io/>
2. Язык командного процессора Cmd. exe операционной системы Windows: учеб. пособие / А. В. Попов. Саранск: Изд-во Мордов. ун-та, 2009. 56 с.
https://andpop.ru/popov_books/cmd.pdf
3. Попов А. В. Командная строка и сценарии Windows. М.: ИНТУИТ, 2016.
<https://intuit.ru/studies/courses/1059/225/info>
4. Tidrow R., Boyce J., Shapiro J.R. Windows 10 Bible. Wiley, 2015. 854 p.
5. Pogue D. Windows 10 May 2019 Update: The Missing Manual. O'Reilly, 2019. 640 p.
6. Лясин, Д. Н. Саньков С. Г. Командные файлы операционной системы MS Windows. Волжский: ВолГГТУ, 2012. 30 с.
<http://window.edu.ru/resource/782/79782>
7. Станек У. Р. Командная строка Microsoft Windows. Справочник администратора. М.: Русская редакция, 2004. 480 с.
8. Книттель Б. Windows 7. Скрипты, автоматизация и командная строка. СПб.: Питер, 2012. 778 с.
9. Wikibooks. Windows Programming/Programming CMD
https://en.wikibooks.org/wiki/Windows_Programming/Programming_CMD
10. Google. Operating Systems and You: Becoming a Power User
<https://www.coursera.org/learn/os-power-user>